Análisis de *Survivable Networks* y Evaluación de la Metodología TRIAD propuesta por el CERT

Autora: Ing. Susana C. Romaniz

Director: Lic. Francisco Javier Díaz

Tesis presentada para obtener el grado de
Magister en Redes de Datos

Facultad de Informática - Universidad Nacional de La Plata

Octubre de 2006

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

SECCION 1: LA CAPACIDAD DE SUPERVIVENCIA EN LOS SISTEMAS EN RED: UNA DISCIPLINA EMERGENTE

En el año 1997, se publica el trabajo "*Survivable Network Systems: An Emerging Discipline*", elaborado por el equipo formado por R. J. Ellison, D. A. Fisher, R. C. Linger, H. F. Lipson, T. Longstaff y N. R. Mead, que constituyó la puesta en escena de este nuevo campo de la ingeniería de desarrollo de software, el cual fue revisado y actualizado en el año 1999. En el mismo se introduce ***la necesidad de replantear un conjunto de conceptos y de metodologías relacionados con los ciclos de vida de los sistemas de información de cara a las nuevas realidades tecnológicas y socios-culturales, que constituyen verdaderos retos para los profesionales e investigadores***.

En una primera etapa se introducen aspectos destacables del trabajo antes mencionado, con miras a una mejor comprensión del proceso seguido en el campo bajo estudio. En el mismo se destacaba que, atendiendo al hecho que la sociedad en su conjunto estaba haciéndose cada vez más dependiente de los sistemas en red a gran escala altamente distribuidos que operan en ambientes de redes ilimitadas –*unbounded*-, tales como la Internet, cuya *principal característica es la de no tener un control administrativo central ni una política de seguridad unificada, y en las que el número y la naturaleza de los nodos conectados a este tipo de redes no puede ser totalmente conocido, y en los que a pesar de los mejores esfuerzos realizados por los profesionales en seguridad, ningún nivel de aseguramiento podía garantizar que un sistema que está conectado a una red ilimitada será invulnerable ante un ataque*.

Por ello, entendían que la ***disciplina de la supervivencia*** podía ayudar a que este tipo de sistemas estuviera en condiciones de proporcionar los servicios esenciales y mantener las propiedades esenciales tales como integridad, confidencialidad y desempeño, ante la presencia de intrusiones, y que, a diferencia de las medidas tradicionales de seguridad que requieren de un control o administración central, *la supervivencia está orientada a hacer frente a los ambientes de red ilimitadas*.

El trabajo describía el enfoque de la capacidad de supervivencia destinada a asistir en las actividades de aseguramiento de un sistema que debe operar en una red ilimitada, de tal manera que el mismo resulte robusto ante la presencia de un ataque y sobreviva a los ataques que resulten de intrusiones exitosas. Además se incluía ***el análisis de la capacidad de supervivencia como un framework integrado a nivel de ingeniería, el estado en que se encontraba de la práctica de la supervivencia, la especificación de requerimientos de supervivencia, las estrategias para alcanzar la supervivencia, y las técnicas y procesos para el análisis de la supervivencia***.

1.1 CAPACIDAD DE SUPERVIVENCIA EN LOS SISTEMAS EN RED

Los actuales sistemas en red de gran escala altamente distribuidos mejoran la eficiencia y la efectividad de las organizaciones al permitir la integración general de la organización en niveles nuevos. No obstante, tal *integración viene acompañada por elevados riesgos de intrusión y compromiso*. Estos *riesgos pueden ser mitigados mediante la incorporación de capacidades de supervivencia dentro de los sistemas de la organización*. Como una disciplina emergente, la supervivencia se apoya en campos de estudio relacionados (por ejemplo, seguridad, tolerancia a fallo, protección, fiabilidad, reutilización, desempeño, verificación, y testeo) e introduce nuevos conceptos y principios. *La supervivencia se focaliza en la preservación de los servicios esenciales en ambientes ilimitados, aún cuando los sistemas en este tipo de ambientes sean penetrados y comprometidos*[Anderson 97].

1.1.1 El nuevo paradigma de red: Integración Organizacional

Desde sus modestos comienzos hace casi 20 años atrás, las redes de computadoras se han vuelto el elemento crítico de la sociedad moderna; no sólo poseen un alcance global, sino que también han impactado sobre prácticamente todas las áreas de la actividad humana, siendo los principales agentes posibilitadores en el campo del comercio, la industria y el gobierno. Los principales sectores de la economía dependen de un vasto arreglo de redes que operan a escalas local, nacional y global. Esta dependencia dominante de la sociedad respecto de las redes amplía las consecuencias de las intrusiones, accidentes y fallas, y aumenta la importancia crítica del aseguramiento de su capacidad de supervivencia.

A medida que las organizaciones intentan mejorar su eficiencia y competitividad, ha ido emergiendo un nuevo paradigma de red, en el que las redes están siendo utilizadas para lograr niveles de integración organizacional radicalmente nuevos, que elimina de raíz los límites de la organización tradicional, y transforma las operaciones locales en componentes de procesos de negocios globales residentes en la red. Por ejemplo, las organizaciones comerciales están incorporando operaciones con unidades de negocio, proveedores y clientes a través de redes de gran escala que mejoran la comunicación y los servicios, combinando operaciones antes fragmentadas y conformando procesos coherentes abiertos a diferentes participantes organizacionales. Este nuevo paradigma representa un salto desde las redes confinadas –*bounded*- con un control central en dirección a las redes ilimitadas. *Las redes ilimitadas se caracterizan por un control administrativo distribuido sin una autoridad central, cuya visibilidad se extiende más allá de los límites de la administración local, y la falta de información completa acerca de la red*. Al mismo tiempo, el grado de dependencia organizacional respecto de las redes está en aumento, y los riesgos y las consecuencias de las intrusiones y compromisos se amplían.

1.1.2 Definición de Supervivencia

Se define a la supervivencia como la capacidad de un sistema de satisfacer su misión, de manera oportuna, ante la presencia de ataques, fallas o accidentes. Se utiliza el término *sistema* en el sentido más amplio posible, incluyendo sistemas de redes y sistemas de gran escala.

El término misión hace referencia a un conjunto de requerimientos de muy alto nivel u objetivos. Cualquier organización o proyecto exitoso debe tener una visión de sus objetivos, ya sea expresados implícitamente o como una declaración formal de su misión. Los juicios acerca de si una misión ha sido satisfecha o no generalmente se hacen dentro del contexto de las condiciones externas que afectan al logro de dicha misión. Por ejemplo, asumiendo que un sistema financiero crítico sale de servicio por 12 horas durante un apagón causado por un huracán; si el sistema preserva la integridad y confidencialidad de sus datos y retoma sus servicios esenciales luego que el período de estrés ambiental ha sido superado, razonablemente se puede juzgar que el sistema ha satisfecho su misión; sin embargo, si el mismo sistema sale de servicio de manera inesperada durante 12 horas bajo condiciones normales (o bajo condiciones de estrés ambiental relativamente menores) y priva a sus usuarios de servicios financieros esenciales, razonablemente se puede juzgar que el sistema ha fracasado en su misión, aún cuando haya sido preservada la integridad y privacidad de sus datos.

La oportunidad es un factor crítico que generalmente se encuentra incluido (o implicado) en los requerimientos de muy alto nivel que definen una misión. No obstante, la oportunidad es un factor tan importante que lo hemos incluido explícitamente en la definición de supervivencia.

Los términos *ataque*, *falla* y *accidente* pretenden incluir todos los eventos potencialmente dañinos; pero estos términos no desagregan estos eventos en conjuntos mutuamente excluyentes o incluso diferenciables. A menudo resulta dificultoso diferenciar si un evento lesivo particular es el resultado de un ataque malicioso, de la falla de un componente, o de un accidente. Aún si la causa puede ser eventualmente determinada, la criticidad de una respuesta inmediata no puede depender de tal conocimiento teórico a futuro.

Los *ataques* son eventos potencialmente dañinos orquestados por un adversario inteligente. Los ataques incluyen intrusiones, sondeos y denegación de servicio. Más aún, la amenaza de un ataque puede tener un impacto tan severo sobre un sistema como si el mismo realmente hubiera ocurrido. Un sistema que asume una posición defensiva debido a la amenaza de un ataque puede reducir su funcionalidad y distraer recursos en actividades de monitoreo del ambiente y protección de los activos del sistema.

Se incluyen las fallas y los accidentes como parte de la supervivencia. Las *fallas* son eventos potencialmente dañinos causados por deficiencias en el sistema o en un elemento externo en el

que depende el sistema. Las fallas pueden ser debidas a errores de diseño del software, degradación del hardware, errores humanos, o datos corrompidos. Los *accidentes* describen un amplio rango de eventos potencialmente dañinos que ocurren de manera azarosa, tales como desastres naturales. Se tiende a pensar en los accidentes como eventos generados externamente (es decir, fuera del sistema) y fallas como eventos generados internamente.

Con respecto a la supervivencia del sistema, una distinción entre falla y accidente es menos importante que el impacto del evento. Ni tampoco resulta posible distinguir entre los ataques orquestados de manera inteligente y eventos perjudiciales que se producen de manera no intencional o azarosa. Por lo general, para que un sistema puede sobrevivir, debe reaccionar frente (y poderse recuperar) un efecto dañino (por ejemplo, queda comprometida la integridad de una base de datos) mucho antes que se pueda identificar la causa fundamental. De hecho, la reacción y recuperación debe resultar exitosa pueda o no determinarse la causa.

El principal finalidad del trabajo es el de ayudar a los sistemas a sobrevivir a los actos de adversarios inteligentes. Esta orientación se basa en la naturaleza de la organización a la que pertenecen los autores. El *Survivable Network Technology Team* es una consecuencia del *CERT® Coordination Center*, el cual ha venido ayudando desde 1988 a usuarios a responder y recuperarse frente a situaciones derivadas de incidentes de seguridad en computadoras.

Finalmente, ***resulta importante reconocer que es el cumplimiento de la misión lo que debe sobrevivir, no algún subsistema o componente particular***. *Resulta esencial para la noción de supervivencia la capacidad de un sistema de satisfacer su misión, aún cuando porciones importantes del mismo se encuentren dañadas o destruidas*. A veces se empleará el término sistema con capacidad de supervivencia como una manera abreviada y algo imprecisa de indicar un sistema con la capacidad de satisfacer una misión específica de cara a ataques, fallas o accidentes. Nuevamente, es su misión, no una porción particular, lo que debe sobrevivir.

1.1.3 El dominio de la supervivencia: Redes Ilimitadas

El éxito de un sistema con capacidad de supervivencia depende del ambiente computacional en el cual opera el mismo; la tendencia irreversible en los ambientes de computación basados en redes está en la dirección de infraestructuras de red ilimitadas. Un sistema confinado es aquél en el cual todas las partes del sistema están controladas por una administración unificada y puede ser completamente caracterizado y controlado; al menos en teoría, el comportamiento de un sistema confinado puede ser comprendido y todas sus partes, identificadas. En un sistema ilimitado no existe un control administrativo unificado sobre todas sus partes; se utiliza el término *control administrativo* en el sentido estricto, el que incluye el poder de imponer y hacer cumplir

® CERT está registrado en *U.S. Patent and Trademark Office*

sanciones, y no simplemente recomendar una política de seguridad apropiada; en un sistema de este tipo cada participante posee una vista incompleta del conjunto, debe depender y confiar en información suministrada por sus vecinos, y no puede ejercer el control fuera de su dominio local.

Un sistema ilimitado puede estar compuesto por sistemas confinados e ilimitados conectados entre sí conformando una red; la Figura 1 muestra un dominio ilimitado constituido por un conjunto de sistemas confinados, en los cuales cada uno de ellos se encuentra bajo un control administrativo separado. Si bien la política de seguridad de un sistema confinado particular no se puede hacer cumplir completamente más allá de los límites de su control administrativo, la política se puede utilizar como una vara de medida para evaluar el estado de seguridad de ese sistema confinado. Por otra parte, la política de seguridad puede ser anunciada en el exterior del sistema confinado; pero los administradores se encuentran seriamente limitados en su capacidad de coaccionar o convencer a personas o entidades externas a seguirla. Esta limitación resulta particularmente verdadera cuando un dominio ilimitado traspasa límites jurisdiccionales, lo que hace difícil o imposible la imposición de sanciones legales.

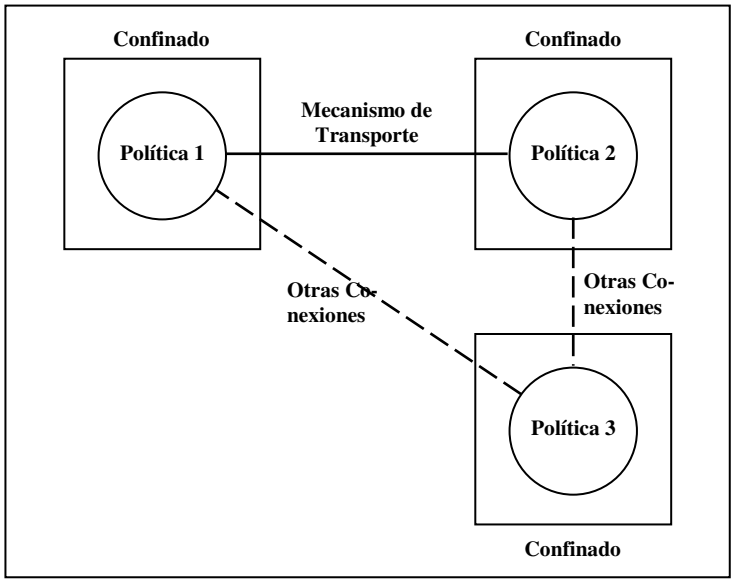


Figura 1: Un dominio ilimitado visto como un conjunto de sistemas confinados

Cuando una aplicación o un sistema *software-intensive*[1] se encuentra expuesto a un ambiente conformado por múltiples e impredecibles dominios administrativos sin límites mensurables, el sistema se encuentra dentro de un ambiente ilimitado. *Un ambiente ilimitado exhibe las siguientes propiedades*:

[1] Un sistema en el que todas las capacidades o comportamientos involucrados importantes se encuentran implementados vía software dentro del bucle de control.

- *Múltiples dominios administrativos sin una autoridad central.*
- *Ausencia de visibilidad global (es decir, el número y la naturaleza de los nodos no se puede conocer por completo).*
- *Interoperabilidad entre dominios administrativos determinada por convención.*
- *Sistemas ampliamente distribuidos e inter-operables.*
- *Usuarios y atacantes pueden ser poseer el mismo rango dentro del ambiente.*
- *No se puede dividir en un número finito de ambientes confinados.*

La Internet es un ejemplo de un ambiente ilimitado con muchas aplicaciones en red cliente-servidor: un servidor Web público y sus clientes pueden estar presente dentro de muchos dominios administrativos diferentes sobre la Internet; no obstante, no existe una autoridad central que exija que todos los clientes se encuentren configurados en la manera esperada por el servidor Web (en particular, un servidor Web nunca puede presuponer la presencia o ausencia de un conjunto de *plug-ins* en el cliente a fin de proveer alguna función).

Para el caso de un servidor Web que soporta una transacción financiera (por ejemplo, una compra basada en Web), el servidor puede requerir que el usuario instale un *plug-in* del lado del cliente para soportar una transacción segura; no obstante, debido a la naturaleza ilimitada del ambiente, *plug-ins* previamente instalados de un competidor pueden corromper, subvertir o dañar al servidor Web durante la transacción. Para que el servidor Web posea capacidad de supervivencia, debe contar con protección incorporada contra interacciones maliciosas del cliente y estas protecciones no deben hacer ninguna suposición respecto a la configuración o características del cliente remoto.

En este ejemplo, el servidor Web y sus clientes constituyen el sistema. Los múltiples dominios administrativos están representando por la diversidad de dominios de sitios sobre la Internet. Muchos de estos dominios poseen usuarios legítimos, mientras que otros sitios se utilizan para efectuar intrusiones de manera anónima. Estos últimos no pueden ser distinguidos por sus dominios administrativos, sino sólo por el comportamiento del cliente. La interoperabilidad entre el servidor y sus clientes se encuentra definida por HTTP (protocolo de transferencia de hipertexto), una convención acordada entre el servidor y los clientes. El sistema, compuesto por los servidores Web y los clientes, se encuentra ampliamente distribuido tanto geográficamente como lógicamente a través de Internet. Usuarios legítimos y atacantes poseen el mismo rango dentro del ambiente, y no existe ningún método para aislar a los usuarios legítimos de los atacantes. En otras palabras, no existe forma de confinar el ambiente a los usuarios legítimos sólo utilizando la política administrativa común.

Los sistemas ilimitados son un componente significativo en el ambiente computacional actual e,

incluso, jugarán un rol mucho mayor en el futuro. La Internet –una red de sistemas no jerárquicos, sólo bajo el control administrativo local- es un el principal ejemplo de un sistema ilimitado. En tanto existen en Internet convenciones que le permiten a los partes trabajar de manera conjunta, no existe un control administrativo global que asegure que estas partes se comporten de acuerdo a estas convenciones. En consecuencia, los problemas de seguridad abundan. *Desafortunadamente, los problemas de seguridad asociados con los sistemas ilimitados generalmente se subestiman*.

1.1.4 Características de los sistemas con capacidad de supervivencia

Una característica clave de los sistemas con supervivencia es su capacidad de entregar servicios esenciales en oportunidad de sufrir ataques, fallas o accidentes.

Fundamental a la posibilidad de entregar servicios esenciales es *la capacidad de un sistema de mantener propiedades esenciales (es decir, niveles especificados de integridad, confidencialidad, desempeño y otros atributos de calidad) en presencia de ataque, falla o accidente*. En consecuencia, *resulta importante definir niveles mínimos de atributos de calidad que deban estar asociados con los servicios esenciales*.

Estos ***atributos de calidad*** *son tan importantes que las definiciones de supervivencia a menudo se expresan en términos del mantenimiento de un balance entre múltiples atributos de calidad*, tales como desempeño, seguridad, fiabilidad, disponibilidad, tolerancia a fallo, modificabilidad y asequibilidad. El proyecto *Architecture Tradeoff Analysis* del *Software Engineering Institute* está empleando esta visión de la supervivencia basada en el balance de atributos (es decir, *tradeoff*) para evaluar y sintetizar sistemas con supervivencia [Kazman 98]. ***Los atributos de calidad representan un vasto número de categorías de requerimientos relacionados, por lo que un atributo de calidad puede contener otros atributos de calidad***. *La capacidad de entregar servicios esenciales (y mantener las propiedades esenciales asociadas) debe se sustentada aún cuando una porción significativa del sistema se encuentre incapacitada. Más aún, esta capacidad no debería ser dependiente de la supervivencia de un recurso de información, cómputo o enlace de comunicaciones específico.*

Resulta clave para el concepto de supervivencia, entonces, la identificación de los servicios esenciales (y de las propiedades esenciales *que los respaldan) dentro de un sistema en operación.* Los ***servicios esenciales*** se definen como *las funciones del sistema que se deben ser mantenidas cuando el ambiente es hostil o se han detectado fallas o accidentes que amenazan al sistema*. Por lo general, existen muchos servicios que pueden temporalmente verse suspendidos cuando un sistema está haciendo frente a un ataque u otra condición ambiental extraordinaria; tal suspensión puede ayudar a aislar áreas afectadas por una intrusión y liberar recursos del sis-

tema que están luchando contra sus efectos. *La función global de un sistema se debería adaptar para preservar los servicios esenciales*.

Propiedad Esencial	Descripción	Ejemplo
Resistencia a ataques	Estrategias para repeler ataques	• Autenticación de usuarios • Diversidad estocástica de programas
Reconocimiento de los ataques y de la extensión del daño	Estrategias para detección de ataques (incluidas intrusiones) y comprensión del estado corriente del sistema, incluida la evaluación de la extensión del daño	• Reconocimiento del empleo de patrones de intrusión • Chequeo de integridad interna
Recuperación completa de los servicios esenciales luego del ataque	Estrategias de restauración de información o funcionalidad comprometidas, que limitan la extensión del daño, el mantenimiento o, si fuera necesario, la restauración de servicios esenciales dentro de las restricciones de tiempo de la misión, restaurando los servicios completos en las condiciones permitidas	• Replicación e reinicialización de los datos
Adaptación y evolución para reducir la efectividad de ataques futuros	Estrategias para mejorar la supervivencia del sistema en base al conocimiento adquirido acerca de intrusiones	• Incorporación de nuevos patrones de reconocimiento de intrusión

Tabla 1. Propiedades esenciales de los sistemas con supervivencia.

Se ha relacionado la capacidad de un sistema con supervivencia en satisfacer su misión de una manera oportuna con su habilidad de entregar servicios esenciales en presencia de ataque, accidente o falla; a fin de cuentas, debe sobrevivir la satisfacción de la misión, no alguna porción o componente del sistema. Si se pierde un servicio esencial, el mismo puede ser reemplazado por otro servicio que soporte la satisfacción de la misión de una manera diferente pero equivalente. No obstante, l*a identificación y protección de servicios esenciales es una parte importante de una metodología práctica para la construcción y análisis de sistemas con supervivencia*. Como un primer resultado, se define que los *servicios esenciales* incluyan conjuntos alternativos a los servicios esenciales (tal vez mutuamente excluyentes) que no necesitan estar disponibles en forma simultánea (por ejemplo, un conjunto de servicios esenciales para soportar la entrega de energía puede incluir la red de distribución de energía eléctrica y la operación de una red de gas natural).

Para mantener sus capacidades de entrega de servicios esenciales, los sistemas con supervi-

vencia deben exhibir las cuatro propiedades claves que se especifican en la Tabla 1.

1.1.5 La supervivencia como un *framework* integrado de ingeniería

Como un paradigma de ingeniería general, la supervivencia es un framework natural para la integración de las disciplinas de ingeniería del software emergentes y aquéllas ya afianzadas al servicio de un objetivo común; entre las áreas afianzadas relacionadas con la supervivencia se incluyen seguridad, tolerancia a fallo, protección, fiabilidad, reutilización, desempeño, verificación y testeo.

1.1.5.1 Supervivencia y seguridad

La disciplina de la seguridad computacional ha realizado valiosas contribuciones a la protección e integridad de los sistemas de información a lo largo de las tres décadas pasadas. No obstante, *la seguridad computacional tradicionalmente ha sido utilizada en términos binarios que sugiere que en todo momento, un sistema se encuentra o seguro o comprometido*. Pero este uso de la seguridad computacional *crea puntos de vista que ignoran, en su mayor parte, los aspectos de la recuperación a partir del compromiso de un sistema y los aspectos relacionados con el mantenimiento de los servicios durante y a posteriori de una intrusión*. Este enfoque es inadecuado para soportar las mejoras necesarias en el estado del arte de la protección de sistemas computacionales frente a un ataque. En contraste, el término *sistemas con supervivencia hace referencia a sistemas cuyos componentes colectivamente llevan a cado su misión aún bajo ataque y a pesar de intrusiones activas que efectivamente dañan una porción significativa del sistema*. La robustez bajo ataque es al menos tan importante como la solidez o resistencia.

La solidez contribuye a la supervivencia, pero la robustez bajo ataque (y, en particular, la recuperabilidad) es la característica esencial que distingue la supervivencia de la seguridad tradicional. Al mismo tiempo, la supervivencia puede beneficiarse de la investigación y la práctica de la seguridad computacional, y puede proveer un *framework* para la integración de la seguridad con otras disciplinas que pueden contribuir con la supervivencia de un sistema.

1.1.5.2 Supervivencia y tolerancia a fallo

La supervivencia requiere robustez bajo condiciones de intrusión, falla o accidente. *El concepto de supervivencia incluye la tolerancia a fallo, pero no es equivalente. La tolerancia a fallo se relaciona con la probabilidad estadística de una falla accidental o la combinación de fallas, no de un ataque malicioso*. Por ejemplo, el análisis de un sistema puede determinar que la ocurrencia simultánea de tres fallas estadísticamente independientes (f_1, f_2 y f_3) podrán hacer que el sistema funcione mal; la probabilidad de que estas tres fallas independientes ocurran, por accidente, en forma simultánea, es extremadamente pequeña, pero un adversario inteligente con

conocimiento de las características internas del sistema puede orquestar su ocurrencia simultánea y sacar de operación al sistema. Lo más probable es que un sistema con tolerancia a fallo no prevea la posibilidad de la ocurrencia de las tres fallas simultáneas, en caso que la probabilidad de ocurrencia se encuentre por debajo de un umbral de consideración; en cambio, un sistema con supervivencia requiere un plan de contingencia para enfrentar tal posibilidad.

La redundancia es otro factor que puede contribuir con la supervivencia de sistemas; no obstante, la redundancia por sí misma resulta insuficiente dado que múltiples sistemas de respaldo idénticos comparten idénticas vulnerabilidades. *Un sistema con supervivencia requiere que cada sistema de respaldo ofrezca funcionalidad equivalente, pero una variación significativa en la implementación*, ya que la variedad estorba los intentos de comprometer al sistema principal y a todos los sistemas de respaldo con una única estrategia de ataque.

1.1.6 La visión del estado del arte en sistemas con supervivencia hacia finales de los '90.

Para esos años resultaba evidente que ***gran parte de la investigación y práctica en el campo de la supervivencia de sistemas computacionales presentaba una vista de la defensa contra las intrusiones peligrosamente limitada y basada en la seguridad***. Esta visión limitada resultaba incompleta debido a estar focalizada casi exclusivamente en el fortalecimiento de un sistema (por ejemplo, empleando tecnología de *firewall* o un método del *orange_book* para la protección del *host*) para evitar su quiebre u otro ataque malicioso, *haciendo poco respecto a cómo detectar una intrusión o qué hacer una vez que ha ocurrido una intrusión y la misma está teniendo lugar*; a su vez, estaba acompañada por *técnicas de evaluación que se limitaban a la fortaleza relativa de un sistema, en oposición a la robustez de un sistema bajo ataque y la aptitud de recuperar sus capacidades comprometidas*.

Se concluía que, si bien la arquitectura de los sistemas seguros confinados dominantes hasta esas años se construye sobre la existencia de una política de seguridad y su forzosa ejecución, impuesta mediante el ejercicio del control administrativo, un sistema ilimitado no posee un control administrativo a través del cual imponer la política de seguridad global (por ejemplo, la arquitectura del *backbone* de la Internet existe independientemente de las consideraciones de la política de seguridad debido a que no hay ningún control administrativo global).

Por otra parte, los sistemas prácticos y asequibles casi nunca están 100% hechos a medida, sino más bien se encuentran construidos a partir de componentes comerciales COTS (de la expresión "*common off-the-shelf*"). *La tendencia hacia el desarrollo de sistemas vía la integración y reutilización en lugar de un diseño hecho a medida y el esfuerzo de su codificación es la piedra angular de la ingeniería de software moderna*. Desafortunadamente, la complejidad intelectual asociada con el diseño, desarrollo y testeo de software virtualmente asegura que los *bugs* explo-

tables en los productos comerciales y de dominio público con estructura internas que están disponibles para su análisis, pueden y serán descubiertos. Cuando estos productos son incorporados como componentes de sistemas más grandes, estos últimos se vuelven vulnerables a estrategias de ataques basadas en estos *bugs* explotables. *Los componentes comerciales de uso difundido y de dominio público ofrecen a los atacantes un omnipresente conjunto de blancos con estructuras internas bien conocidas y, por lo general, invariantes; la falta de variabilidad entre los componentes se traduce en una falta de variabilidad entre sistemas, por lo que éstos permiten, potencialmente, que una única estrategia de ataque posea un impacto de gran alcance y devastador*.

La natural escalada de las amenazas ofensivas versus las contramedidas defensivas ha demostrado hace tiempo y en forma reiterada que no puede ser construido ningún sistema práctico que sea invulnerable a ataques. A pesar de los mejores esfuerzos, no existe certeza de que esos sistemas no sean quebrados. En consecuencia, se proponía que la visión tradicional de la seguridad de los sistemas de información debiese ser expandida para abarcar una especificación y diseño del comportamiento del sistema tal que ayude a la supervivencia del sistema a pesar de las intrusiones activas; sólo entonces sería posible crear sistemas que resulten robustos en presencia de un ataque y capaces de superar aquellos ataques que no pueden ser completamente repelidos.

La naturaleza del desarrollo corriente de sistemas estipula que aún los sistemas más fuertes pueden y serán quebrados; en consecuencia, la supervivencia debería estar diseñada como parte de los sistemas para ayudar a evitar los potenciales efectos devastadores que comprometen y hacen fallar al sistema debido a la intrusión.

1.1.6.1 El manejo de incidentes mejora la supervivencia

Si bien la aplicación del término supervivencia a los sistemas de computación resultaba relativamente nueva, la práctica de la supervivencia no lo era, ya que mucha de la práctica de supervivencia se había venido dando con los equipos de respuesta a incidentes. El *CERT® Coordination Center* (CERT/CC)[2] había venido ofreciendo, a lo largo de su historia, prácticas para la mejora de la supervivencia de sistemas en la comunidad de Internet, proporcionando servicios de respuesta ante incidentes, y publicando y distribuyendo avisos de vulnerabilidades, teniendo particular éxito con su ayuda a sitios indicando medidas relacionadas con la mitigación y recuperación de riesgos.

Su experiencia ha demostrado que la manera en que las organizaciones responden y se recupe-

[2] CERT y CERT Coordination Center están registradas en la U.S. Patent and Trademark Office

ran frente a intrusiones es, al menos, tan importante como los pasos que se siguen para prevenirlos. Ya se pensaba que la amplia disponibilidad y el uso de sistemas con supervivencia por parte de la comunidad de Internet y a lo largo de la infraestructura de Internet constituía la mejor esperanza que se produjeran las mejoras necesarias para transformar a la Internet en un sistema de sistemas de información en red con supervivencia, ayudando así a hacer de la Internet un medio viable para conducir el comercio, la defensa y el gobierno.

También se destacaba que muy poco de la tecnología básica en ingeniería en seguridad e integración de sistemas se aplicaba a los sistemas ilimitados; más bien, las prácticas corrientes asumían la capacidad de identificar, definir y caracterizar la extensión del control administrativo sobre un sistema, todos los puntos de acceso al mismo, y todas las señales que pueden aparecer en dichos puntos de acceso; pero en los sistemas ilimitados, tales como la actual Internet, estas condiciones de borde no pueden ser completamente determinadas.

El estado del arte existente en la evaluación de supervivencia y seguridad tendía a tratar a los sistemas y sus ambientes como estáticos e invariables; sin embargo, la supervivencia y la seguridad de los sistemas se degrada a lo largo del tiempo a medida que se producen cambios en sus estructuras, configuraciones y ambientes, y a medida que el conocimiento acerca de sus vulnerabilidades se esparce a lo largo de la comunidad de intrusos.

En aquellos años, la piedra angular de la seguridad en la Internet era un *firewall*, un sistema lógicamente confinado dentro de uno ilimitado físicamente. Los autores sostenían que pensar en sistemas confinados dentro de dominios ilimitados conducía a diseños y arquitecturas de seguridad deficientes desde la perspectiva de la supervivencia[3]. La adición de componentes activos, tales como capacidad de detección y respuesta dinámica, permitiría que los *firewalls* jugasen su rol en los sistemas con supervivencia.

1.2 DEFINICIÓN DE LOS REQUERIMIENTOS

Los requerimientos de supervivencia pueden variar muy sustancialmente dependiendo del alcance del sistema, su criticidad y las consecuencias debidas a la falla o interrupción del servicio. Las categorías de las definiciones de requerimientos para los sistemas con supervivencia incluyen función, uso, desarrollo, operación y evolución. A continuación se presentan definiciones de requerimientos de supervivencia, maneras en las cuales se pueden expresar estos requerimientos, y su impacto sobre la supervivencia del sistema. ***El nuevo paradigma para la***

[3] Un ejemplo notable era el uso de un *firewall* como el componente básico de la seguridad para la Internet. Este método es severamente limitado y puede ser fácilmente soslayado mediante la explotación de diferencias fundamentales entre sistemas confinados e ilimitados. Los *firewalls stateless* representan el estado del arte para las arquitecturas de seguridad, pero no para sistemas con supervivencia, debido a que eran dispositivos pasivos de sólo filtrado.

definición y diseño de los requerimientos de sistema está caracterizado por servicios distribuidos, lógica distribuida, código distribuido (incluido contenido ejecutable), hardware distribuido, una infraestructura compartida de comunicaciones y encaminamiento, confianza disminuida, y la falta de un control administrativo unificado. *El aseguramiento de la supervivencia de los sistemas de misión crítica desarrollados bajo este nuevo paradigma es un formidable esfuerzo de alto riesgo para la ingeniería de desarrollo de software*. Este esfuerzo requiere que las medidas de seguridad informática tradicional sean ampliadas por nuevas estrategias globales de supervivencia del sistema.

1.2.1 Expresión de los requerimientos de supervivencia

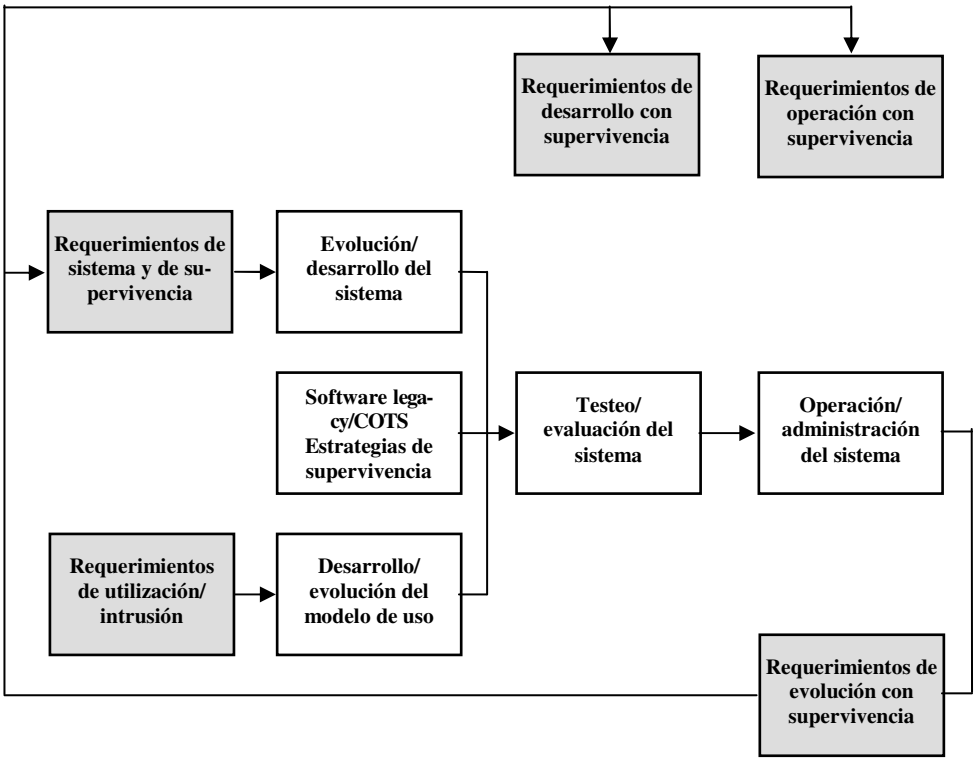


Figura 2. Definición de requerimientos para sistemas con supervivencia

La definición y el análisis de los requerimientos de supervivencia es un primer paso crítico en el logro de un sistema con supervivencia [Linger 98]. La Figura 2 muestra un modelo iterativo para la definición de estos requerimientos. *La supervivencia no sólo debe contemplar los requerimientos referidos a la funcionalidad del software, sino también los requerimientos relativos al uso, desarrollo, operación y evolución del software*. Por lo tanto, en este modelo resultan relevantes para los sistemas con supervivencia cinco tipos de definiciones de requerimientos, que se discuten en detalle a continuación.

Requerimientos de sistema y de supervivencia. El término *requerimientos de sistema* hace

referencia a las funciones tradicionales a nivel de usuario que el sistema debe proveer (por ejemplo, un sistema de administración de red debe proveer funciones que le permitan a los usuarios realizar las operaciones de monitoreo de la red y ajustar los parámetros de desempeño) y a no-funcionales del sistema (tales como, sincronización, desempeño y fiabilidad). El término *requerimientos de supervivencia* hace referencia a las capacidades de un sistema de entregar los servicios esenciales en presencia de intrusiones y compromisos y de recuperar los servicios a pleno. La Figura 3 describe la integración de los requerimientos de supervivencia con los requerimientos de sistema a niveles de nodo y de red.

La supervivencia requiere que los requerimientos del sistema se encuentren organizados en servicios esenciales y no-esenciales. Los servicios esenciales deben ser mantenidos aún durante intrusiones exitosas, mientras que los servicios no-esenciales se recuperan luego que han sido controladas las intrusiones. Estos últimos pueden estar estratificados en un cierto número de niveles, cada uno de los cuales materializan una menor cantidad de servicios cada vez más vitales a medida que la severidad y la duración de la intrusión se incrementan.

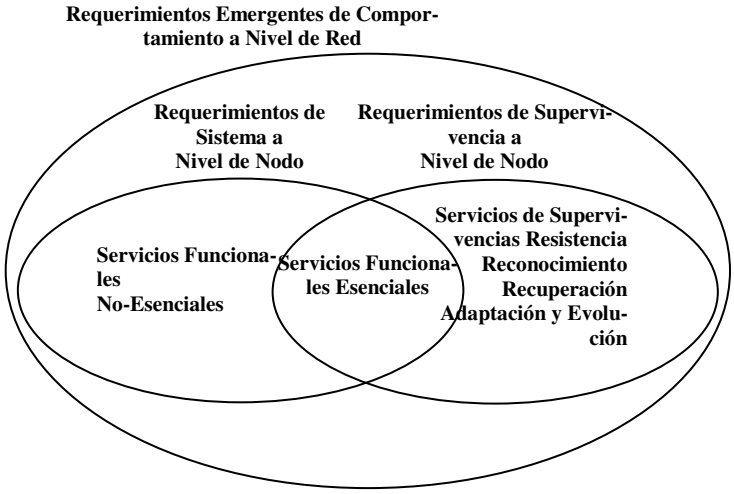


Figura 3. Integración de los requerimientos de supervivencia con los requerimientos del sistema.

Como se mostrara en la Figura 2, los sistemas con supervivencia también pueden incluir componentes *legacy* y COTS que no fueron desarrollados teniendo a la capacidad de supervivencia como un objetivo explícito. Tales componentes pueden proveer tanto servicios esenciales como no-esenciales, y pueden demandar requerimientos funcionales para su aislamiento y control mediante el empleo de *wrappers* y filtros que permitan su uso seguro dentro de un ambiente de sistema con supervivencia.

La Figura 3 muestra que la *supervivencia por sí misma impone nuevos tipos de requerimientos al sistema*. Estos nuevos requerimientos incluyen la *resistencia* a, el *reconocimiento* y la *recuperación* luego de intrusiones y compromisos, y la *adaptación y evolución* para disminuir la

efectividad de futuros intentos de intrusión. Estos requerimientos de supervivencia están soportados por una variedad de estrategias existentes y emergentes, como se indica en la Figura 2 y que luego se discuten con más detalle.

Finalmente, la Figura 4 describe los *requerimientos emergentes del comportamiento* a nivel de red. *Estos requerimientos se caracterizan como emergentes debido a que no están asociados con nodos particulares, sino que emergen a partir del comportamiento colectivo de los servicios de nodo en la comunicación a lo largo de la red*. Atienden a la supervivencia de las capacidades de la red global (por ejemplo, capacidades de encaminar mensajes entre el conjunto de nodos críticos a pesar de cómo las intrusiones puedan dañar o comprometer la topología de la red). Vislumbramos sistemas con supervivencia que estén en condiciones de adaptar su comportamiento, función y asignación de recursos en función de las intrusiones; por ejemplo, cuando es necesario, las funciones y recursos dedicados a servicios no-esenciales podrían ser reasignados a proveer servicios esenciales, y a la resistencia y el reconocimiento y recuperación. Los requerimientos para tales sistemas también deben especificar de qué manera el sistema se debería adaptar y reconfigurar a sí mismo en respuesta a las intrusiones.

Los sistemas pueden exhibir significativas variaciones en los requerimientos de supervivencia. Pequeñas redes locales pueden requerir pocos o ningún servicio esencial y tiempos de recuperación medidos en horas; por el contrario, las redes de redes de gran escala pueden requerir de un conjunto básico de servicios esenciales, detección de intrusión automatizada y tiempos de recuperación medidos en minutos. Los sistemas de comando y control embebidos pueden requerir que los servicios esenciales sean mantenidos en tiempo real y los tiempos de recuperación se midan en milisegundos.

La consecución y mantenimiento de la supervivencia consumen recursos en el desarrollo, operación y evolución de un sistema. La asignación de recursos a la supervivencia del sistema se debería basar en los costos y los riesgos para una organización asociados con la pérdida de servicios esenciales.

Requerimientos de utilización/intrusión. Las pruebas de supervivencia de un sistema deben demostrar tanto el correcto desempeño de los servicios esenciales y no-esenciales de un sistema, como así también la supervivencia de los servicios esenciales bajo intrusión. *Debido a que el desempeño del sistema bajo testeo (y operación) depende totalmente de la utilización del sistema, un método efectivo para testear sistemas con capacidad de supervivencia se basa en el empleo de escenarios derivados de modelos de utilización* [Mills 92, Trammel 95].

Los modelos de utilización se desarrollan a partir de requerimientos de utilización, que especifican los ambientes y escenarios de uso del sistema. Los requerimientos de utilización para los

servicios esenciales y no-esenciales deben estar definidos en paralelo con los requerimientos del sistema y de supervivencia. Más aún, los intrusos y los usuarios legítimos se deben considerar de manera equivalente. También se deben definir los requerimientos de intrusión que especifican los ambientes y escenarios de utilización/intrusión. Con esta aproximación, la utilización de los servicios del sistema por parte de los intrusos y de los usuarios legítimos se modela en forma conjunta.

La Figura 4 muestra la relación entre utilización legítima y bajo intrusión. Los intrusos no sólo podrían intervenir en escenarios que están más allá se los escenarios legítimos, sino también en usos legítimos con propósitos de la intrusión en caso que los mismos adquieran los privilegios necesarios.

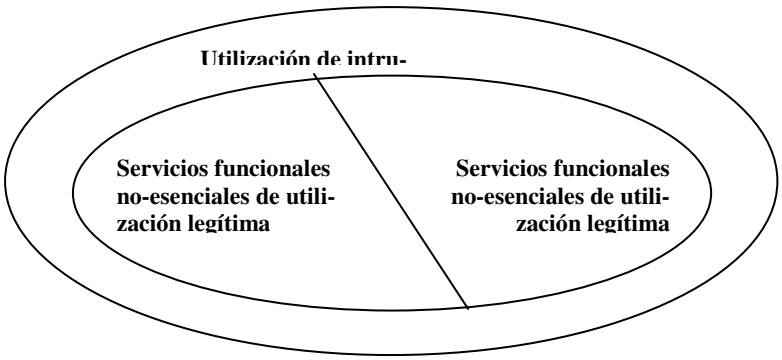


Figura 4. Relación entre utilización legítima y bajo intrusión.

Requerimientos de desarrollo con supervivencia. La supervivencia impone requerimientos rigurosos sobre las prácticas de desarrollo y testeo del sistema. *Una funcionalidad inadecuada y errores de software pueden tener un efecto devastador sobre la supervivencia del sistema, y proporcionar oportunidades para su explotación por parte de un intruso. Se requieren de sólidas prácticas de ingeniería para la creación de software con capacidad de supervivencia*.

Los siguientes cinco principios (cuatro técnicos y uno organizacional) son requerimientos ejemplo para las prácticas de desarrollo y testeo de sistemas con supervivencia:

- *Especificar de manera precisa las funciones requeridas por el sistema en todas las posibles circunstancias de utilización del sistema.*
- *Verificar la exactitud de las implementaciones del sistema con respecto a las especificaciones funcionales.*
- *Especificar la función de utilización en todas las posibles circunstancias de uso del sistema, incluido el uso del intruso.*
- *Testear y certificar el sistema en base a la función de utilización y a métodos estadísticos.*
- *Crear ágiles equipos de trabajo permanentes para el monitoreo, adaptación y evolu-*

ción del sistema.

Asimismo, se requieren de sólidas prácticas de ingeniería para tratar con los componentes de software legacy y COTS.

Requerimientos de operación con supervivencia. *La supervivencia impone demandas sobre los requerimientos de operación y administración del sistema*, entre los que se incluyen la definición y comunicación de las políticas de supervivencia, el monitoreo del uso del sistema, la respuesta ante intrusiones, y la evolución de las funciones del sistema según se necesite asegurar la supervivencia a medida que cambian los ambientes de utilización y los patrones de intrusión a lo largo del tiempo.

Requerimientos de evolución con supervivencia. La evolución del sistema responde a los requerimientos del usuario por nuevas funcionalidades. No obstante, *esta evolución también resulta necesaria para responder al creciente conocimiento con que cuentan los intrusos respecto del comportamiento y estructura del sistema*. En particular, *la supervivencia requiere que las capacidades del sistema evolucionen más rápidamente que el conocimiento de un intruso*. Esta rápida evolución evita que los intrusos acumulen información acerca del comportamiento invariante del sistema sobre el que ellos necesitan conseguir una exitosa penetración y explotación.

1.2.1.1 Definición de Requerimientos para los Servicios Esenciales

En la discusión precedente se hizo la distinción entre servicios esenciales y no-esenciales. *Cada requerimiento del sistema se debe examinar para determinar si corresponde a un servicio esencial. El conjunto de servicios esenciales debe conformar un subsistema viable para los usuarios que sea completo y coherente. En caso que se necesiten de varios niveles de servicios esenciales, cada conjunto de servicios provistos en cada nivel también deben ser examinados en lo que hace a su integridad y coherencia. Asimismo, se deben definir requerimientos en lo que hace a la transición entre niveles de servicios esenciales*.

Al hacer la distinción entre servicios esenciales y no-esenciales, se pueden aplicar todos los procesos y métodos usuales de definición de requerimientos. Las técnicas de elicitación tales como aquéllas encarnadas en la Ingeniería de Requerimientos del Software pueden ayudar a identificar los servicios esenciales [Ebert 97]. Se debe trabajar sobre el análisis y equilibrio del costo/beneficio para determinar los conjuntos de servicios que atienden de manera suficiente los riesgos y vulnerabilidades de la supervivencia del negocio; asimismo, se deben establecer criterios para el seguimiento de los requerimientos de supervivencia a lo largo del diseño, desarrollo y testeo. Como se mencionara anteriormente, la simulación de la intrusión a través de escenarios de utilización/intrusión está incluida en el proceso de testeo.

1.2.1.2 Definición de Requerimientos para los Servicios con Capacidad de Supervivencia

Luego de especificar los requerimientos para los servicios esenciales y no-esenciales, se deben *definir un conjunto de requerimientos para los servicios con supervivencia*, los que pueden estar organizados dentro de cuatro categorías: resistencia, reconocimiento, recuperación, y adaptación y evolución. Los mismos deben operar dentro de un *ambiente de intrusión que se puede caracterizar vía tres fases diferentes de intrusión: penetración, exploración y explotación*.

Fase de Penetración. En esta fase, un intruso intenta ganar acceso a un sistema a través de varios escenarios de ataque. Estos escenarios varían desde entradas al azar realizadas por *hackers* aficionados hasta ataques bien planeados efectuados por intrusos profesionales. Estos intentos están diseñados para capitalizar vulnerabilidades conocidas del sistema.

Fase de exploración. En esta fase, el intruso ha ganado acceso a facilidades de un sistema deseado, y ha realizado operaciones destinadas a comprometer las capacidades del sistema.

Fase de explotación. En esta fase, el intruso ha ganado acceso a facilidades deseadas del sistema y se encuentra realizando operaciones destinadas a comprometer las capacidades del sistema.

Penetración, exploración y explotación crean una espiral de autoridad creciente del intruso y un círculo de compromiso que se ensancha. Por ejemplo, la penetración en el nivel de usuario generalmente es un medio para encontrar vulnerabilidades a nivel de administrador; luego, se emplea la autorización a nivel de usuario para explotar estas vulnerabilidades para lograr una penetración a nivel de administrador; finalmente, el compromiso del *host* más débil dentro de un sistema en red permite que ese *host* sea utilizado como trampolín para comprometer otros *hosts* más protegidos.

Las definiciones de requerimientos para los servicios de resistencia, reconocimiento, recuperación, y adaptación y evolución ayudan a seleccionar las estrategias de supervivencia relacionadas con estas fases de intrusión.

Requerimientos de resistencia. *La resistencia es la capacidad de un sistema de impedir ataques*. En consecuencia, la resistencia es importante durante las fases de penetración y exploración, antes de la explotación real. Las estrategias actuales para lograr resistencia incluyen el uso de *firewalls*, autenticación y encriptación. *La diversificación es una estrategia de resistencia que seguramente se volverá más importante en el caso de redes ilimitadas*.

Los requerimientos de diversificación deben definir la variación planificada de la función, estructura y organización en un sistema con supervivencia, y los medios para alcanzarlas. La diversificación está orientada a crear un *objetivo en movimiento*, y tornar poco efectiva la acumulación de conocimiento sobre sistema como estrategia de intrusión. También elimina las oportunidades de

intrusión asociadas con múltiples nodos que ejecutan idéntico software y generalmente exhiben vulnerabilidades idénticas. Tales sistemas ofrecen una economía de escala tentadora para los intrusos, debido a que cuando un nodo ha sido penetrado, todos los nodos pueden serlo. Los requerimientos de diversificación pueden incluir variación en los programas, los datos recolectados, y el encaminamiento y comunicación de red. Por ejemplo, se pueden definir medios sistemáticos para aleatorizar los programas de software preservando la funcionalidad [Linger 99a].

Requerimientos de reconocimiento. *El reconocimiento es la capacidad de un sistema de identificar ataques o pruebas que preceden al ataque*. La habilidad de reaccionar o adaptarse durante una intrusión es esencial para la capacidad de un sistema de sobrevivir a un ataque que no puede ser completamente repelido. Para reaccionar o adaptarse, el sistema primero debe reconocer que está siendo atacado. De hecho, *el reconocimiento resulta esencial en las tres fases de ataque*.

Las actuales estrategias para el reconocimiento de ataques incluyen la detección de intrusión en su estado corriente, y las técnicas tanto de *logging* y auditoría como de investigación de seguimiento de reportes generados por mecanismos de detección de errores ordinarios. Las técnicas de detección de intrusión avanzadas generalmente son de dos tipos: *detección de anormalidades* y *reconocimiento de patrones*. La primera de ellas se basa en modelos de comportamientos normales del usuario. A menudo estos modelos se establecen a través del análisis estadístico de los patrones de uso. Las desviaciones respecto de estos patrones de uso son señaladas como sospechosas. El reconocimiento de patrones se basa en modelos de comportamiento del intruso. Las actividades de un usuario que se corresponde con un patrón conocido de comportamiento de intruso levantan una alarma.

Los requerimientos de las futuras redes con supervivencia probablemente emplearán estrategias adicionales tales como *autoconciencia*, *mantenimiento de la confianza*, y *reporte de caja negra*.

Autoconciencia es le proceso de establecer un modelo semántico de alto nivel de los cómputos que un componente o sistema se encuentra ejecutando y que le ha sido solicitado que ejecute. Un sistema o componente que comprende qué es lo que le está siendo solicitado puede rechazar solicitudes que podrían resultar peligrosas, comprometer una política de seguridad, o impactar de manera adversa en la entrega de los servicios mínimos esenciales.

El mantenimiento de la confianza es alcanzado por un sistema mediante periódicas peticiones entre sus componentes (por ejemplo, entre los nodos de una red) para testear y evaluar continuamente las relaciones de confianza. La detección de signos de intrusión accionará un inmediato testeo de las relaciones de confianza.

El reporte de caja negra es un *dump* de información del sistema que se puede recuperar luego de la falla de un sistema o un componente para su análisis y determinación de la causa de dicha

falla (por ejemplo, un error de diseño o un tipo específico de intrusión). Este análisis puede ayudar a evitar que otros componentes sufran el mismo destino.

El diseño de un sistema con capacidad de supervivencia debe incluir requerimientos explícitos de reconocimiento de ataque. Estos requerimientos aseguran el uso de una o más de las estrategias precedentes mediante la especificación de características arquitectónicas, herramientas automatizadas, y manuales de procesos. *Dado que las técnicas de intrusión están constantemente avanzando, los requerimientos de reconocimiento deberán ser frecuentemente revisados y mejorados en forma continua*.

Requerimientos de recuperación. *La recuperación es la capacidad de un sistema de restaurar los servicios luego de ocurrida una intrusión*; también *contribuye con la capacidad de un sistema de mantener los servicios esenciales durante la intrusión*.

Los requerimientos relacionados con la recuperación son los que más claramente distinguen a los sistemas con supervivencia de los sistemas que son simplemente seguros. La seguridad informática tradicional guía el diseño de los sistemas que se basan casi completamente en la fortaleza (es decir, la resistencia) de la protección. Una vez que la seguridad ha sido quebrada, el daño puede continuar sin encontrar mucha resistencia en el camino. La habilidad de un sistema de reaccionar durante una intrusión activa resulta central para su capacidad de sobrevivir a un ataque que no puede ser completamente repelido. En consecuencia, la recuperación es crucial durante las fases de exploración y explotación de una intrusión.

Las estrategias de recuperación en uso hoy en día incluyen *la replicación de información y servicios críticos, el empleo de diseños tolerantes a fallo, y la incorporación de sistemas de respaldo para el hardware y el software*. Estos sistemas de respaldo mantienen copias del software crítico aisladas de la red. Algunos sistemas, tales como sistemas de procesamiento transaccional de gran escala, emplean elaborados procesos *roll-back* de transacción de fina granulometría para mantener la consistencia e integridad de los datos.

Requerimientos de adaptación y evolución. *La adaptación y evolución son críticos para el mantenimiento de la resistencia al siempre creciente conocimiento de los intrusos sobre cómo explotar las funciones invariantes del sistema*. La permanente adaptación dinámica mejora la habilidad del sistema de resistir, reconocer y recuperarse frente a intentos de intrusión. Por ejemplo, un requerimiento de adaptación puede ser una infraestructura que le permita al sistema vacunarse a sí mismo contra vulnerabilidades de seguridad recientemente descubiertas mediante la distribución y aplicación de *fixes* de seguridad a todos los elementos de la red. Otro requerimiento de adaptación puede ser que el conjunto de reglas de detección de intrusión se actualice regularmente en respuesta a reportes de actividades de intruso conocidas obtenidas de fuentes

autorizadas de información de seguridad, tal como el *CERT® Coordination Center*.

Los requerimientos de adaptación aseguran que tales capacidades son una parte integral del diseño de un sistema. Como con los casos de requerimientos de resistencia, reconocimiento y recuperación, la constante evolución de las técnicas de intrusión demandan que los requerimientos de adaptación sean frecuentemente revisados y mejorados de forma continua.

1.3 ESTRATEGIAS DE DISEÑO E IMPLEMENTACION

A continuación se examinan estrategias que dan soporte a la capacidad de supervivencia de las funciones críticas del sistema en redes ilimitadas. *Las estrategias de supervivencia dentro de los sistemas en red dependen de varios supuestos y restricciones*. Si bien éstos pueden parecer obvios, estos supuestos y restricciones se deben hacer explícitos. *Los supuestos difieren radicalmente de los supuestos implícitos tradicionalmente hechos para los sistemas basados en uniprocesador, multiprocesador y en red confinada, en los que se han estado basando la investigación y el desarrollo hasta el momento*.

Para las redes ilimitadas, se asume que:

- *Cualquier nodo individual de la red puede ser comprometido*.
- *La supervivencia no requiere que ningún componente físico particular de la red sea preservado*.
- *Sólo los servicios esenciales de la red en su conjunto deben sobrevivir*.
- *Por razones de fiabilidad, error de diseño, error de usuario, y compromiso intencional, la confianza en un nodo de la red o cualquier nodo con el cual el mismo se puede comunicar no puede ser garantizada*.

Atendiendo a que las redes ilimitadas cuentan con las tres características principales que están presentes en cada definición: la falta de un control central físico o administrativo, la ausencia de una comprensión o visión de todas las partes de la red, y no presentar un límite práctico de crecimiento en el número de nodos de la red, estas características imponen las siguientes restricciones sobre la arquitectura de las redes con capacidad de supervivencia y sobre el formato de las estrategias de supervivencia posibles:

- *No existe un único punto de falla dentro de la red*. Los servicios esenciales se encuentran distribuidos de una manera que no depende de manera crítica de ningún componente o nodo particular.
- *La comprensión global resulta imposible de alcanzar dentro de un sistema distribuido*. En su lugar, están definidos protocolos para la interacción y el conocimiento compartido entre los nodos.

- *Cada nodo debe validar en forma continua su propia fiabilidad y la de aquéllos con los que él se comunica*.
- *El esfuerzo computacional en un nodo dado de una red ilimitada*, para ya sea los servicios esenciales, la comunicación, o la validación de confianza, *debe tener niveles que se reduzcan de manera proporcional al número de nodos dentro de la red*.

1.3.1 Cuatro aspectos de las Estrategias para la solución de supervivencia

Como se introdujera anteriormente, existen cuatro aspectos de la solución de supervivencia que pueden servir como base para las estrategias de supervivencia. Estos cuatro aspectos son: resistencia, reconocimiento, recuperación, y adaptación y evolución del sistema. A continuación se resumen las metodologías para cada una de estas cuatro áreas.

Existen muchas técnicas que tienen que ver con estos cuatro aspectos, algunas o todas de ellas se pueden aplicar a los sistemas con supervivencia. No se listan todas, sino que se las categoriza a partir de aspectos más amplios. La Tabla 2 contiene los cuatro aspectos de la solución de supervivencia y taxonomías representativas de las respectivas estrategias.

1.3.2 Soporte de las Estrategias por parte de la Infraestructura Computacional

El rápido crecimiento de la Web y otras aplicaciones basadas en Internet han alentado el desarrollo de una infraestructura computacional de soporte a las aplicaciones distribuidas. En tanto que los esfuerzos iniciales en la Web estuvieron concentrados en la publicación de información, su dominio de aplicación se ha extendido hasta alcanzar a espectro mucho más amplio de las necesidades computacionales de una organización. El foco técnico de este crecimiento se ha desplazado desde herramientas tales como navegadores o servidores Web hacia el desarrollo de un conjunto de servicios provistos comercialmente a través de Internet y compatibles con la misma. Ejemplos de estos servicios son servicios de archivos, impresión, transacciones, mensajería, directorios seguridad y objetos, tales como CORBA (*Common Object Request Broker Architecture*) y DCOM (*Distribuited Component Object Model*).

Las infraestructuras distribuidas disponibles comercialmente se encuentran en una fase temprana de su desarrollo y todavía no proporcionan directamente capacidades de supervivencia. El reconocimiento no es un servicio soportado y la recuperación es provista en forma indirecta por un servidor de transacciones. Generalmente, una organización adopta tal infraestructura a costos más bajos mediante el empleo de una infraestructura común para aplicaciones disponibles vía *intranets*, *extranets* e Internet, y para simplificar el desarrollo de aplicaciones embebiendo la complejidad de la computación distribuida dentro de la infraestructura más que en cada aplicación.

La administración de datos de perfil de usuario es un ejemplo de un servicio que puede asumir

una infraestructura distribuida. Un requerimiento general de la supervivencia del sistema es la de atender a la autenticación de usuario, y la administración de la autoridad otorgada a ese usuario respecto del acceso a datos y a sistemas. La autenticación puede ser implementada empleando contraseñas y autorizaciones que son validadas mediante listas de control de acceso. No obstante, en muchos sistemas, tales como aplicaciones de base de datos, las listas de control de acceso están soportadas por la misma aplicación.

Aspecto de supervivencia	Taxonomías de estrategias
Resistencia	• Seguridad tradicional, incluida la encriptación y los canales encubiertos • Diversidad y maximización de diferencias entre nodos particulares • *Analytic redundancy* y *Voting*[4] • Validación continua de la confianza • Exhibición de propiedades estocásticas y comportamiento aleatorio
Reconocimiento	• *Analytic redundancy* y testeo (incluidas fallas en software, encriptación y confianza) • Monitoreo de intrusión y actividades sospechosas • Monitoreo del comportamiento e integridad del sistema
Recuperación	• Redundancia física y de información • Copias no-locales de recursos de información • Preparación, presteza, planificación de contingencias y equipos de respuesta
Adaptación y Evolución	• Cambios generales y específicos para resistir, reconocer, o recuperarse ante nuevas vulnerabilidades que van siendo descubiertas • Difusión de alertas a todos los demás nodos • Difusión de estrategias de adaptación y evolución • Disuasión vía represalia o castigo

Tabla 2. Una taxonomía de las estrategias relacionadas con la supervivencia

Cuando los usuarios, los datos y las aplicaciones del sistema se encuentran distribuidos geográficamente, se vuelve dificultoso mantener dentro de la aplicación los datos del perfil de usuario. Un servicio de directorio compartido, el cual forma parte de una infraestructura distribuida,

[4] Las técnicas más difundidas para implementar sistemas de alta disponibilidad, ya sean de hardware o de software, incluyen la replicación con *mayority voting* y *analytic redundancy*. La replicación con *mayority voting* requiere de un conjunto de módulos idénticos, y determina la aquella salida que ha de ser de la mayoría de los módulos. Este método también es conocido como *N-version programming* cuando se lo emplea para proveer confiabilidad a un software. *Analytic redundancy* emplea un sistema de respaldo menos complejo para reemplazar un módulo de alto desempeño en caso de falla. Estos métodos vienen bien para ofrecer tolerancia a fallo dentro de una variedad de configuraciones. Sin embargo, en el caso de redes de computadoras, resultan suficientes métodos menos complejos y menos costosos.

puede proporcionar la capacidad de almacenamiento de datos y un protocolo, tal como LDAP (*Lightweight Directory Access Protocol*), para el acceso y reemplazo de los mecanismos de control de acceso específicos de cada aplicación. Estos servicios de seguridad de la infraestructura pueden proporcionar los mecanismos de autenticación de usuario tales como una interfase de clave pública, mecanismos para describir el control de acceso, y los medios para definir una política de seguridad. El uso de servicios compartidos para la autenticación y autorización de usuarios debería reducir la complejidad de la aplicación y del sistema en su conjunto, como así también proveer el medio para definir una política de seguridad de la organización.

Cuando se implementa esta estrategia, la arquitectura del sistema está constreñida por los servicios provistos por la infraestructura y los protocolos soportados. Por ejemplo, una estrategia de supervivencia puede ser el intercambio de un servicio principal con una implementación alternativa de ese servicio para el caso que el primero se vea comprometido. En el actual estadio de despliegue de la infraestructura, si bien existe un cierto grado de interoperabilidad entre los servicios provistos por diferentes fabricantes, presenta una integración de servicios tal que hace difícil o imposible el reemplazo de un servicio, como por ejemplo un servicio de directorio, con otro de un proveedor diferente.

El empleo de servicios de directorio compartido también plantea problemas generales de supervivencia. Una infraestructura utilizada de manera extendida debería desarrollar un robusto conjunto de servicios. Sin embargo, su difundido empleo da lugar a una importante e informada comunidad de intrusos y a una vasta diseminación de información acerca de las vulnerabilidades del sistema y de las soluciones de seguridad. Un directorio comprometido o inaccesible puede afectar a múltiples aplicaciones y a múltiples sitios.

Una parte esencial para atender a la capacidad de supervivencia de un sistema es el establecimiento de procedimientos operacionales y administrativos para el sistema de directorios de tal manera que los administradores de sistema puedan monitorear el servicio y ofrecer recuperación. El balance en el diseño es que la implementación de procedimientos de monitoreo y recuperación sea menos costosa al momento de emplear componentes compartidos que emplear la arquitectura específica de una aplicación. Los servicios de infraestructura proveen un soporte genérico de replicación y mantenimiento de la consistencia a través de sitios distribuidos. No obstante, lograr la supervivencia de la misión global requiere no sólo del conocimiento del impacto de los datos de control de acceso comprometidos y del diseño de una política de recuperación, sino también del conocimiento de las aplicaciones del sistema.

Los productos de infraestructura disponibles comercialmente proporcionan servicios generales que son independientes del dominio de aplicación. Sin embargo, algunos de los servicios listados en la Figura 3 requieren de conocimientos específicos del dominio de aplicación. Por ejem-

plo, el reconocimiento de una intrusión o el mantenimiento de la confianza entre los nodos requiere del conocimiento del comportamiento esperado. Un protocolo puede asegurar que la información sea entregada, pero no puede validar la pertinencia de los datos. Se pueden incluir mecanismos de recuperación simples en los *logs* de transacciones o de restauración de archivos; pero el empleo de transacciones, estrategias *rollback* y técnicas más avanzadas requieren de experiencia en el dominio para identificar estados consistentes de la aplicación y el impacto de los datos comprometidos.

El empleo exitoso de tales técnicas de recuperación se ha dado en productos centrados en la aplicación, tales como sistemas de base de datos relacionales que administran estructuras de datos relativamente homogéneas. La aplicación de estas técnicas a sistemas de computación distribuida generales es más dificultosa.

1.3.3 Consideraciones en el Diseño de la supervivencia

Se pueden esbozar un cierto número de consideraciones acerca de las preguntas y problemas que deben ser tenidos en cuenta relacionados con la supervivencia del sistema en los sistemas en red.

1.3.3.1 La Supervivencia requiere del mantenimiento de la Confianza

Un tema abierto es de qué manera determinar la base de confianza y cómo un nodo particular de una red contribuye a la supervivencia de los servicios esenciales del sistema cuando:

- Un nodo cualquiera se puede tornar no confiable
- No existe una visión global o un control global
- Los nodos no pueden confiar completamente en sí mismos ni en sus vecinos

Dependiendo de la aplicación, esto puede ser posible mediante el diseño arquitectónico o la acción dinámica dentro sistema destinado a incrementar la fiabilidad, visibilidad y control de los componentes o la responsabilidad de los participantes. No obstante, la única base absoluta del mantenimiento de la confianza es la consistencia en la retroalimentación del comportamiento obtenida a partir de las interacciones con otros nodos, y en forma independiente de la verificación de las acciones indicadas por nodos que no están directamente involucrados con las transacciones.

Un aspecto estrechamente relacionado es la ausencia de una visión y un control global. Si se determina la presencia de componentes no fiables en un sistema, puede resultar extremadamente difícil determinar si las funciones críticas han sido comprometidas o no sin contar con una visión y un control global. Si están ausentes la visión y el control global (y en general esto es lo que sucede) esta condición impide el diseño de efectivas arquitecturas basadas en red con capacidad de supervivencia. En particular, debería ser posible que los nodos particulares generalmente contribuyan con los objetivos de supervivencia y, al menos, no interfieran con estos objetivos.

Los algoritmos genéticos[5], por ejemplo, logran estos efectos a través de la acción colectiva de los participantes individuales. Sin embargo, estos participantes no pueden medir la efectividad global ni determinar si su contribución es positiva. Este ejemplo sugiere que las soluciones de supervivencia pueden estar presentes entre los algoritmos emergentes que dependen de la continua interacción con sus nodos vecinos, pero que no requieren retroalimentación indicando el progreso y el éxito [Fisher 99].

1.3.3.2 El Análisis de la Supervivencia Basado-en-Protocolo, no en-Topología

Otra implicancia de los sistemas basados en red es que los aspectos importantes de su arquitectura desde el punto de vista de la supervivencia se relacionan con las convenciones y reglas de interacción entre sus nodos vecinos, y que la topología de la red es en gran parte irrelevante. Esto es, las arquitecturas de red deben ser especificadas, comparadas, y medidas en términos de sus interacciones y no de la topología de su interconexión.

Como un ejemplo de este tipo de análisis, consideremos que el problema general de la persistencia de los datos de estado correspondientes a un protocolo. ¿Un protocolo debería mantener información de estado para mejorar la fiabilidad o realizar chequeos de consistencia adicionales? ¿Qué nivel de chequeo debería soportar la infraestructura? J. H. Saltzer y sus colegas han examinado el FTP (*File Transfer Protocol*) y comparado enfoques que chequean paquetes sólo en los nodos origen y destino (extremo-a-extremo) con protocolos que chequean fiabilidad en cada salto del camino de comunicaciones [Saltzer 84]. La conclusión fue que el chequeo salto-a-salto incrementa la complejidad y afecta el desempeño con un reducido incremento en la fiabilidad global.

Kenneth P. Birman analiza estos puntos de balance en un contexto más general [Birman 96]. Propiedades tales como fiabilidad y supervivencia pueden ser mejoradas mediante propiedades que soportan tolerancia a falla o garantizan la comunicación. Sin embargo, el costo de una propiedad que soporte, digamos, ordenamiento uniforme de eventos, puede resultar miles de veces superior que una propiedad más débil que pueda requerir que la aplicación maneje un comportamiento no-uniforme.

Argumentos similares se pueden realizar cuando comparamos arquitecturas *stateless* o datos no-replicados para atender a un fuerte requerimiento de consistencia a nivel de aplicación. En el caso de arquitecturas *stateless* y datos no-replicados, el servidor puede volver a arrancar y los clientes tener la responsabilidad de reconectarse. La supervivencia requiere de un análisis para balancear las responsabilidades entre los servidores y los clientes, y entre el monitoreo extremo-

[5] El algoritmo genético es una modelo de máquina de aprendizaje que deriva su comportamiento de una metáfora de los procesos de evolución en la naturaleza. Esto se lleva a cabo mediante la creación dentro de una máquina de una población de individuos, representados por cromosomas (en esencia un conjunto de cadenas de caracteres) , análogos a los cromosomas base-4 que se observan en nuestro ADN. Luego los individuos de la población llevan a cabo un proceso de evolución.

a-extremo realizado por la aplicación y el monitoreo del protocolo provisto por la infraestructura. Para tal estrategia de recuperación, el nivel de aplicación puede ser el nivel apropiado en el cual analizar el comportamiento del estado de la aplicación y del usuario, y seleccionar las acciones de recuperación apropiadas.

1.3.3.3 La Supervivencia es Emergente y Estocástica

Los objetivos de supervivencia son propiedades emergentes deseables para el sistema en su conjunto, pero no necesariamente predominantes en los nodos particulares del sistema. Esta aproximación contrasta con los diseños tradicionales en los cuales las funciones o propiedades especializadas son aseguradas en nodos particulares, y la composición del sistema debe asegurar que estas propiedades y capacidades funcionales sean preservadas en el sistema como un todo. A los fines de la supervivencia, debemos lograr propiedades que abarquen al sistema que generalmente no existen en los nodos particulares. Un sistema con capacidad de supervivencia debe asegurar que las propiedades de supervivencia deseadas emergen de las interacciones entre los componentes durante la construcción de sistemas fiables a partir de componentes poco fiables.

La supervivencia es inherentemente estocástica. Si las propiedades de supervivencia son emergentes, sólo estarán presentes en un sistema cuando el número de nodos que contribuyen a las mismas es lo suficientemente grande. Si el número u ordenamiento de los nodos cae por debajo de un umbral crítico, se fracasa en acompañar a la propiedad de supervivencia. Un ejemplo de este tipo de propiedad crítica de supervivencia en la conectividad en las comunicaciones del sistema.

Podemos diseñar la arquitectura del sistema para maximizar el número de caminos entre dos nodos cualesquiera; pero si han sido comprometidos una cantidad suficientes de enlaces como para particionar la red, la comunicación entre nodos arbitrarios ya no será tendrá éxito. En consecuencia, las propiedades, algoritmos y arquitecturas de supervivencia deberían estar especificados, visualizados y evaluados para determinar la probabilidad de su éxito bajo condiciones dadas de uso y no determinadas como entidades discretas.

1.3.3.4 La Supervivencia requiere de un Componente de Administración

El diseño de un sistema con capacidad de supervivencia también incluye la gestión de operación y administración. Una pobre administración de sistema es la causa frecuente de vulnerabilidades en los sitios administrados en forma centralizada. En los sistemas de red ilimitados, la administración del sistema debe estar coordinada a través de múltiples sitios. Los procedimientos de administración de sistema existentes generalmente asumen un ambiente confinado y el completo control administrativo sobre los servicios requeridos. La complejidad de una infraestructura y el uso de servicios externos por fuera del control inmediato de la organización requieren de la

expansión de los servicios administrativos y la provisión de una función de monitoreo como parte de la infraestructura.

1.4 LINEAS DE INVESTIGACIÓN

En este trabajo inicial, ya quedaron planteadas un conjunto de de áreas de investigación prometedoras en los sistemas con capacidad de supervivencia que el equipo de *Survivable Networks Technology* del *Software Engineering Institute* se plantearon como abiertas:

- Técnicas de descripción de la adaptación y desarrollo arquitectónico para describir adecuadamente sistemas distribuidos de gran escala con atributos de supervivencia.
- Representación de ambientes de intrusión mediante modelos utilizados por el intruso.
- Creación de un método de análisis para evaluar la capacidad de supervivencia como una propiedad emergente global a partir de la especificación arquitectónica.
- Refinamiento de la tecnología e instrumentos de análisis a través de testeos pilotos de sistemas distribuidos reales.

SECCION 2: EL DESARROLLO DE UN MÉTODO

Continuando con la línea de trabajo iniciado en el año 1997 con la publicación de "*Survivable Network Systems: An Emerging Discipline*", el equipo de N. R. Mead, R. J. Ellison, R. C. Linger, T. Longstaff y J. McHugh publica en el año 2000 "*Survivable Network Analysis Method*" -Método SNA-, desarrollado por el CERT® *Coordination Center* del *Software Engineering Institute*.

Atendiendo a que la supervivencia es la capacidad de un sistema en satisfacer su misión en el momento adecuado a pesar de intrusiones, fallas o accidentes, y que los tres principios de la supervivencia son (1) resistencia a intrusiones, (2) reconocimiento de los efectos de la intrusión, y (3) recuperación de los servicios a pesar del éxito de las intrusiones, entonces *la supervivencia de los sistemas existentes o proyectados se puede analizar en el nivel de la arquitectura o de los requerimientos del sistema*.

Los cuatro pasos del Método SNA guían a las partes implicadas a través de un proceso de análisis destinado a mejorar la supervivencia del sistema cuando un sistema es amenazado. El método se centraliza en la preservación de los servicios esenciales del sistema que soporta la misión de la organización. Las conclusiones del Método SNA se resumen en un Mapa de Supervivencia, que enumera las estrategias arquitectónicas actuales y recomendadas.

Los autores destacaban que el Método SNA había estado siendo aplicado con éxito en sistemas comerciales y gubernamentales, y que continuaba evolucionado hacia una aplicación más rigurosa.

Su publicación es un reflejo de los importantes avances logrados en el afianzamiento de la disciplina de la capacidad de supervivencia en sistemas ilimitados y la presencia de una metodología tendiente ser soporte de la ingeniería de la supervivencia.

2.1 METODO *SURVIVABLE NETWORK ANALYSIS*

Los actuales modelos de ciclo de vida de desarrollo de software no están focalizados en la creación de sistemas con capacidad de supervivencia, y exhiben deficiencias cuando se los utiliza para desarrollar sistemas que están destinados a poseer un alto grado de aseguramiento de la supervivencia [Marmor-Squires 88]. En caso de ser abordados, los problemas de supervivencia a menudo quedan relegados a una amenaza separada de la actividad del proyecto, dando por resultado que la supervivencia sea tratada como una propiedad agregada. Este aislamiento de las consideraciones de supervivencia de las principales tareas de desarrollo del sistema da por resultado una desafortunada postergación en su atención. *La supervivencia debería estar integrada y*

tratada a un mismo nivel que las otras propiedades del sistema, a los fines de desarrollar sistemas que tengan la funcionalidad y el desempeño requeridos, pero que también soporten fallas y compromisos. Importantes decisiones de diseño y de balanceo se tornan más difíciles cuando la supervivencia no está integrada dentro del principal ciclo de vida del desarrollo. La separación de las amenazas de las actividades resulta una labor costosa e intensiva, provocando a menudo una duplicación en el esfuerzo de diseño y documentación. Asimismo, las herramientas de soporte de la ingeniería de supervivencia a menudo no están integradas en el ambiente de desarrollo de software. Con esta separación, se torna más difícil encarar adecuadamente los problemas de alto-riesgo de la supervivencia y las consecuencias de las fallas. Además, las tecnologías que soportan los objetivos de supervivencia, tales como especificación formal, métodos de balanceo de arquitectura, análisis de intrusión, y patrones de diseño de supervivencia, no son aplicadas efectivamente en el proceso de desarrollo.

Para cada actividad del ciclo de vida, se deberían encarar los objetivos, e incorporar los métodos que aseguren la supervivencia. En algunos casos, los métodos de desarrollo existentes pueden mejorar la supervivencia. La actual investigación está creando métodos nuevos que se pueden aplicar; sin embargo, se requiere de mucha más investigación y experimentación antes que los objetivos de la supervivencia puedan volverse una realidad. *Este trabajo describe los conceptos de supervivencia, discute un modelo de ciclo de vida de desarrollo de software para la supervivencia, y describe una técnica que puede ser aplicada durante las actividades de requerimientos, especificación y arquitectura para soportar objetivos de supervivencia*.

2.2 DEFINICION DE LA CAPACIDAD DE SUPERVIVENCIA

Reproduciendo los conceptos expresados en *1.1.1 "El nuevo paradigma de red"* y en *1.1.2 "Definición de Supervivencia"*, destaca que la investigación en el campo de los sistemas con capacidad de supervivencia a lo largo de los últimos años ha dado por resultado el desarrollo de los conceptos y definiciones de supervivencia que se describen en esta sección, los cuales han sido el resultado del trabajo del equipo *Survivable Network Technology* del *Software Engineering Institute* y del *CERT Coordination Center* (CERT/CC).

Asimismo, indica que el *principal objetivo del trabajo es el de proporcionar a los administradores métodos que ayuden a los sistemas a sobrevivir frente a actos de adversarios inteligentes. En tanto tienen puestos su foco en las intrusiones, los métodos analizados aplican en su totalidad tanto a fallas como a accidentes*.

2.2.1 Características de los sistemas con capacidad de supervivencia

Retomando los principales aspectos y conceptos presentados en *1.1.4 "Características de los*

sistemas con capacidad de supervivencia" en lo referente a que la característica clave de los sistemas con supervivencia es su capacidad de entregar servicios esenciales en oportunidad de sufrir ataques, fallas o accidentes, se plantea que, entonces, resulta fundamental para el concepto de supervivencia l*a identificación de los servicios esenciales (y de las propiedades esenciales que los respaldan) dentro de un sistema operacional*.

Los servicios esenciales se definen como las funciones del sistema que se deben ser mantenidas cuando el ambiente es hostil o se han detectado fallas o accidentes que amenazan al sistema. Para mantener sus capacidades de entregar los servicios esenciales, los sistemas con supervivencia deben exhibir las cuatro propiedades claves: *resistencia*, *reconocimiento*, *recuperación* (las tres Rs), y *adaptación*; las mismas se describen en la Tabla 3, cabiendo destacar la actualización respecto del contenido de la Tabla 1.

Propiedad Clave	Descripción	Ejemplos de Estrategia
Resistencia a ataques	Estrategias para repeler ataques	• Autenticación • Controles de acceso • Encriptación • Filtrado de mensajes • *Wrappers* de supervivencia • Diversificación del sistema • Aislamiento funcional
Reconocimiento de ataques y daños	Estrategias para detección de ataques y evaluación de la extensión del daño	• Detección de intrusión • Chequeo de integridad
Recuperación de los servicios esenciales y completa luego del ataque	Estrategias para limitar el daño, restaurar información o funcionalidad comprometida, mantener o restaurar servicios esenciales dentro de las restricciones de tiempo de la misión, restaurar los servicios completos	• Redundancia de componentes • Replicación de datos • Respaldo y restauración del sistema • Plan de contingencia
Adaptación y evolución para reducir la efectividad de ataques futuros	Estrategias para mejorar la supervivencia del sistema en base al conocimiento adquirido acerca de intrusiones	• Nuevos patrones de reconocimiento de intrusión

Tabla 3. Propiedades de los sistemas con supervivencia considerados en el Método SNA.

La Tabla 3 identifica un conjunto de *estrategias de supervivencia que se pueden aplicar para contrarrestar las amenazas de un ataque sobre un sistema*. Algunas de estas técnicas para mejorar la supervivencia están tomadas de otras áreas, principalmente de las comunidades de la seguridad, la protección y la tolerancia a fallo.

En el área de ***resistencia a ataques***, se dispone de un conjunto de técnicas:

- Los mecanismos de autenticación de usuario limitan el acceso a un sistema a un grupo aprobado de usuarios; desde simples contraseñas hasta una combinación de contraseñas, *tokens* de autenticación manejados por el usuario (protegidos ellos mismos mediante contraseñas), y biométricas.
- Los controles de acceso se pueden aplicar al acceso al sistema, o a programas y datos particulares. Cuando se los impone a través de un sistema operativo confiable, aplican automáticamente una política predefinida para el otorgamiento o denegación de acceso a un usuario autenticado. Estos controles pueden servir como un sustituto de los mecanismos basados en contraseñas a nivel de datos o de programas.
- La encriptación puede proteger datos, ya sea dentro de un sistema o en tránsito entre sistemas, de la intercepción o captura física. Las tecnologías disponibles de encriptación son lo suficientemente fuertes como para resistir todos los actuales ataques de fuerza bruta posibles hoy en día. La encriptación transforma el problema de proteger grandes cantidades de datos en un problema de administrar cantidades relativamente pequeñas de material en clave. La encriptación también se puede utilizar para proveer autenticación, no-repudiación, chequeo de integridad, y una variedad de otras propiedades de aseguramiento.
- El filtrado de mensajes generalmente se utiliza en la frontera de un sistema, o parte del mismo, para restringir el tráfico que ingresa al sistema; por ejemplo, no existe razón por la que permitir el ingreso a una instalación de mensajes relacionados con servicios no soportados o indeseados, o mensajes que parecieran se originados desde dentro de la organización probablemente no son legítimos en caso que provengan desde el exterior por lo que no deberían ser permitidos. Asimismo, los filtros pueden estar diseñados para bloquear mensajes asociados con ataques conocidos.
- Los *wrappers* de supervivencia son esencialmente filtros de mensajes aplicados a nivel de la interfase del sistema operativo. Pueden ser utilizados para proporcionar el operador *operand cheking* o para redirigir *calls* a rutinas de librerías no seguras hacia versiones más robustas, y también para imponer una política de control de acceso restrictiva sobre una aplicación en particular.
- La diversificación del sistema combinada con implementaciones redundantes hacen más difícil el trabajo del atacante. En una implementación heterogénea, es probable que un escenario utilizado para atacar una implementación ha de fallar en otras. *Defensive coding*[6] se utiliza para proteger a los programa contra valores de entrada no-válidos. El aislamiento funcional reduce o elimina las dependencias entre los servicios de más po-

[6] Dependiendo del contexto, puede significar diferentes cosas: 1) el acto de insertar en un programa *error-checks* redundantes, para asegurar, en lo posible, que los errores sean capturados en forma temprana y lo más próximo posible de donde realmente ocurrieran; 2) evitar el empleo de constructores que se conoce presentan problemas.

sible. Esto evita que el ataque a un servicio comprometa a otros.

- El aislamiento a veces no es fácil de lograr, dado que las dependencias entre los servicios a veces no son obvias si se las mira desde un nivel de abstracción equivocado. Los servicios que comparten un procesador, por ejemplo. Son mutuamente dependientes uno del otro en lo que hace a recursos como CPU y memoria. Ellos también comparten espacio en disco y probablemente un adaptador de red. Es posible que un proceso dispare un ataque de denegación de servicio hacia otro al monopolizar cualquier de esos recursos. El aislamiento de recurso puede requerir de un mecanismo de compartición basado en *quota* o alguna técnica similar. El aislamiento funcional puede extenderse hasta la separación física de las funciones del sistema, a menudo vía servidores separados sin conexiones lógicas –por ejemplo, separando el procesamiento de *emails* de archivos de datos sensibles. Ningún método electrónico puede saltar una distancia física o penetrar un equipo que está apagado.

En el área del ***reconocimiento de ataques y daños***, existe un número limitado de opciones.

- Generalmente los sistemas de detección de intrusión intentan identificar los ataques ya sea mediante la búsqueda de patrones conocidos de ataques o el uso de un modelo de referencia del comportamiento normal del sistema tratando las desviaciones como potenciales ataques. Ambas técnicas se pueden aplicar tanto al tráfico de red como a datos específicos de una aplicación o de una plataforma. Los *logs* de auditoría del sistema o de una aplicación son la fuente de información para la detección de intrusiones a nivel de la plataforma o la aplicación. Existen sistemas de detección de intrusión en tiempo real y de post-procesamiento. En la actualidad, los IDSs pueden perder muchas intrusiones, especialmente en ataques nuevos, y padecen de altas tasas de falsas alarmas.
- Los verificadores de integridad (*integrity checkers*) pueden detectar intrusiones que modifican archivos o datos del sistema que deberían mantenerse sin cambios; el proceso de verificación implica la creación de un modelo de línea de base de los archivos a ser protegidos utilizando códigos de verificación o firmas criptográficas, y periódicamente comparar el modelo actual con el de la línea de base.

En términos de ***recuperación***, cuando se ha reconocido un ataque dañino (u otra falla), resulta necesario seguir las etapas para la inmediata recuperación de los servicios esenciales y, eventualmente, los servicios completos. Existen una cantidad de técnicas que pueden ser utilizadas, cuyos efectos varían desde el mantenimiento de los servicios completo de manera transparente sin una interrupción apreciable hasta posiciones de último recurso que sólo mantienen un núcleo de los servicios esenciales.

- La redundancia es la clave para el mantenimiento de los servicios completos de cara a fallas. La comunidad dedicada a las soluciones de tolerancia a fallo posee una considerable experiencia en el empleo de la redundancia para mantener un servicio durante la fase de falla de componentes, pero sus técnicas analíticas se basan en el conocimiento de distribuciones estadísticas asociadas con varios mecanismos de falla, algo que puede no resultar posible con las fallas inducidas por ataques.
- En muchos casos, la replicación de datos críticos es un medio de primer orden para lograr la recuperación; cuando los servicios esenciales están siendo provistos por bases de datos de uso difundido, puede resultar posible restaurar los servicios críticos de datos en una ubicación más o menos arbitraria.
- El respaldo sistemático de todos los recursos de datos, combinado con los mecanismos apropiados para su restauración en la plataforma de origen u otra equivalente, es una parte clave para cualquier estrategia de recuperación. La granularidad de los respaldos deberá depender de la frecuencia con que cambian los datos y el costo de repetir el trabajo realizado entre dos respaldos consecutivos. En casos extremos, puede resultar necesario respaldar los archivos cada vez que se los cierra luego de una operación de escritura, y registrar las transacciones o pulsaciones de tecla de tal manera que se puede recuperar el trabajo intermedio. En otros casos, pueden resultar suficientes los respaldos diarios o semanales. Cuando un sistema se encuentra bajo ataque o ha experimentado una falla, puede resultar posible reconfigurar dinámicamente el sistema para transferir los servicios esenciales desde el componente atacado a uno operativo, eliminando los servicios menos esenciales durante dicho proceso; debido a que esta estrategia no posee capacidad redundante, la reconfiguración puede persistir sólo por períodos limitados, dado que la criticidad de los servicios menos esenciales se incrementa con el intervalo de tiempo que los mismos no están disponibles.
- Finalmente, puede resultar posible poner el sistema en un modo de operación alternativo, quizá en uno en el cual el rol del sistema computacional esté temporalmente reducido o incluso eliminado. Por ejemplo, las transacciones computadora-a-computadora podrían ser reemplazadas con faxes enviados manualmente. Un sistema computarizado de inventario y órdenes podría revertirse durante un corto período de tiempo en el sistema manual anterior.

Tal vez la parte más laboriosa de la supervivencia es la adaptar un sistema para hacerlo más robusto en la esperanza de que el mismo resistirá ataques o intrusiones antes nunca vistos. Así como los atacantes están constantemente buscando nuevos puntos de vulnerabilidad, los defen-

sores deben crear defensas que están basadas en generalizaciones de ataques previamente vistos, en un esfuerzo de anticipar las direcciones en los que podrían ocurrir nuevos ataques.

2.3 DEFINICION DEL CICLO DE VIDA DE SUPERVIVENCIA

2.3.1 El modelo en espiral

A continuación se describe un Modelo de Ciclo de Vida que fuera desarrollado para sistemas confiables (*trusted*) [Marmor-Squires 89], y que resulta adecuado para el desarrollo de sistemas con capacidad de supervivencia; *está basado en los modelos en cascada y en espiral, y una extensión del modelo en espiral para incorporar conceptos de sistemas confiables*.

Las características fundamentales del modelo en espiral son la administración del riesgo, la robustez y la flexibilidad. La descripción del modelo en espiral se desprende particularmente de Boehm [Boehm 89]. Seguidamente se describe el modelo en espiral básico y a una especialización del mismo.

El desarrollo de software es, en el mejor de los casos, un proceso dificultoso. De hecho, muchos sistemas de software, especialmente en el área comercial, simplemente evolucionan sin un proceso de desarrollo bien definido; otros son desarrollados utilizando una ambigua progresión de etapas, posiblemente con algún grado de retroalimentación entre etapas adyacentes –por ejemplo, en el modelo en cascada [Royce 87], el cual más bien realiza una explicación del proceso de desarrollo de facto a posteriori más que ofrecer una guía para su ejecución [Parnas 86].

A lo largo de los años, se han propuesto numerosas variaciones o alternativas al modelo en cascada; cada una de estas alternativas han superado ciertos defectos de dicho modelo, pero introducen sus propios problemas. En tanto este modelo atiende al importante propósito de introducir una disciplina en el proceso de desarrollo de software, esencialmente impone progresiones lineales, las que fueran necesarias en el mundo orientado al procesamiento por lotes (*batch*) de limitadas alternativas y escaso potencia computacional, asumiendo un sistema de ensamblado lineal tipo fabril, en el cual comprendemos cada parte.

En la actualidad, la disponibilidad de estaciones de trabajo, redes y almacenamiento masivo barato, junto con una variedad de herramientas, hace posible *una amplia variedad de actividades de programación exploratorias como parte del proceso de desarrollo*. Esto significa que resulta posible desarrollar prototipos o modelos como partes de sistemas, obtener la reacción de una potencial comunidad de usuarios, y retro-alimentar con esta información el proceso de desarrollo. Asimismo, la creciente disponibilidad de ambientes de desarrollo y ejecución han acelerado esta tendencia.

El modelo en espiral es un intento de ofrecer un framework ordenado para el desarrollo de

software que supere las deficiencias del modelo en cascada, y adecue actividades tales como prototipado, reutilización, y codificación automática como una parte del proceso. Una consecuencia de la flexibilidad del modelo de ciclo de vida es que el desarrollador se enfrenta con opciones de muchas etapas de proceso, y con las opciones deviene el riesgo. En consecuencia, gran parte del énfasis en el modelo en espiral está puesto en la administración del riesgo. Esto, a su vez, puede resultar en un progreso desparejo en varios aspectos del desarrollo del sistema, con áreas de alto riesgo que están siendo exploradas en profundidad en tanto que se difieren áreas de bajo riesgo.

El modelo en espiral ve al proceso de desarrollo en un sistema de coordenadas polares. La coordenada *r* representa el costo del proyecto acumulativo y la coordenada *w* representa la fecha de avance. *El plano se encuentra dividido en cuatro cuadrantes que representan los diferentes tipos de actividades*, tal como se indican:

I Determinación de objetivos, alternativa y restricciones

II Evaluación de alternativas; identificación y resolución de riesgos

III Actividades de desarrollo

IV Revisión y planificación de ciclos futuros.

Además, la frontera entre los cuadrantes I y IV representa el compromiso de adelantar con un elemento, solución o método particular, y avanzar hacia la siguiente etapa (o espiral) dentro de un espacio definido de actividades (por ejemplo, diseño). Actividades específicas pueden superponer múltiples espirales. Además, se pueden requerir espirales concurrentes para atender áreas de riesgo diverso. La línea de compromiso puede implicar una decisión de terminar el proyecto o de cambiar la dirección en base a los resultados obtenidos.

La Figura 5 muestra un ciclo de la espiral. Debemos observar que *w* no avanza de manera uniforme con el tiempo. Algunos ciclos de la espiral pueden requerir de meses para completarse, mientras que otros sólo requieren de días; de manera similar, si bien el incremento de *w* indica avance dentro de un ciclo de la espiral, no necesariamente denota un progreso hacia la terminación del proyecto.

Cada ciclo del modelo atiende a todas las actividades entre los eventos de revisión y compromiso. En el inicio del proceso, los ciclos pueden ser cortos en tanto se exploran alternativas en el espacio de decisión del proyecto. A medida que se van resolviendo los riesgos, los ciclos se pueden alargar, de tal manera que el cuadrante de desarrollo incluye varias etapas de la cascada. La espiral se puede finalizar con la entrega del producto –en cuyo caso las actividades de modificación o mantenimiento son nuevas espirales- o continuar hasta que el producto es abandonado.

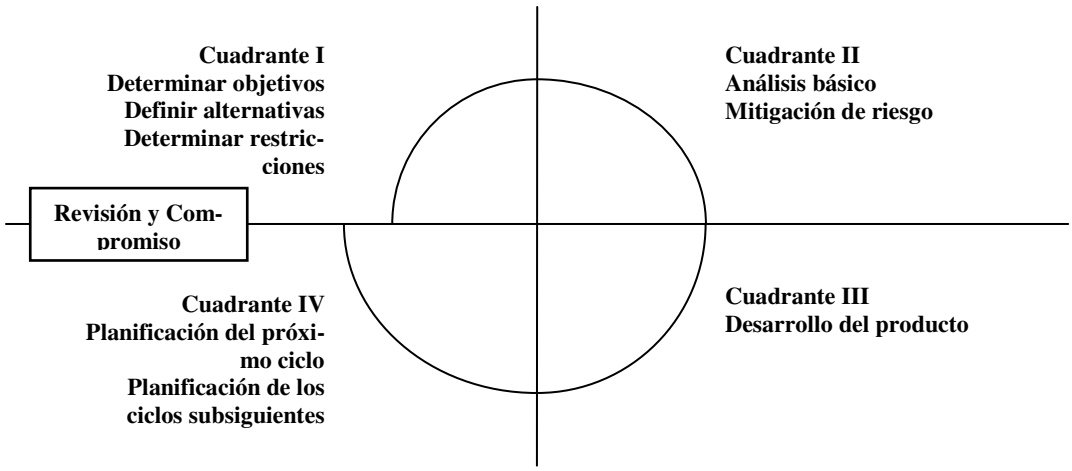


Figura 5. Ciclo en espiral de un proyecto.

2.3.2 Un modelo en espiral para el desarrollo de sistemas con capacidad de supervivencia

El proceso en espiral "puro" generalizado recién discutido proporciona un *framework* para modelos más especializados. *La especialización y el mejoramiento consisten en la adaptación de las actividades llevadas a cabo conforme al modelo a los requerimientos especiales* de los sistemas que están siendo producidos.

Esto se lleva a cabo mediante *la especificación de (1) las actividades que se hacen cargo de los conductores que caracterizan al sistema y (2) las restricciones que caracterizan el ambiente en el cual el sistema está siendo producido. Esta combinación da lugar a una versión especializada del modelo en espiral que integra la capacidad de supervivencia dentro del proceso de administración*, como se describe en la Figura 6.

Los sistemas con capacidad de supervivencia deben satisfacer una variedad de intereses conflictivos:

- *Los usuarios finales necesitan que los sistemas lleven a cabo su misión operativa principal, posiblemente a expensas de violar políticas de seguridad bajo ciertas circunstancias.*
- *Resulta común el caso en que los sistemas deban también satisfacer alguna autoridad de certificación o acreditación; las etapas requeridas para estas aprobaciones pueden entrar en conflicto con los intereses de los usuarios.*
- *Los desarrolladores desean terminar el trabajo, preferentemente antes de lo planificado y por debajo de lo presupuestado.*
- *Dentro de la organización encargada del desarrollo, pueden existir tensiones entre los diferentes especialistas involucrados.*
- *La resolución de estos conflictos puede traducirse en restricciones en el ambiente y en proceso de desarrollo.*

- *Siempre están presentes las consideraciones de costo. El proceso de desarrollo en espiral ha demostrado ser el más efectivo en costo que los métodos tradicionales, pero exhibe una diferente distribución del costo a lo largo del tiempo. Bajo el modelo en espiral, los desembolsos generalmente son mayores en las actividades de especificación y diseño iniciales, lo que conlleva a una reducción de costos en las actividades de implementación e integración posteriores.*

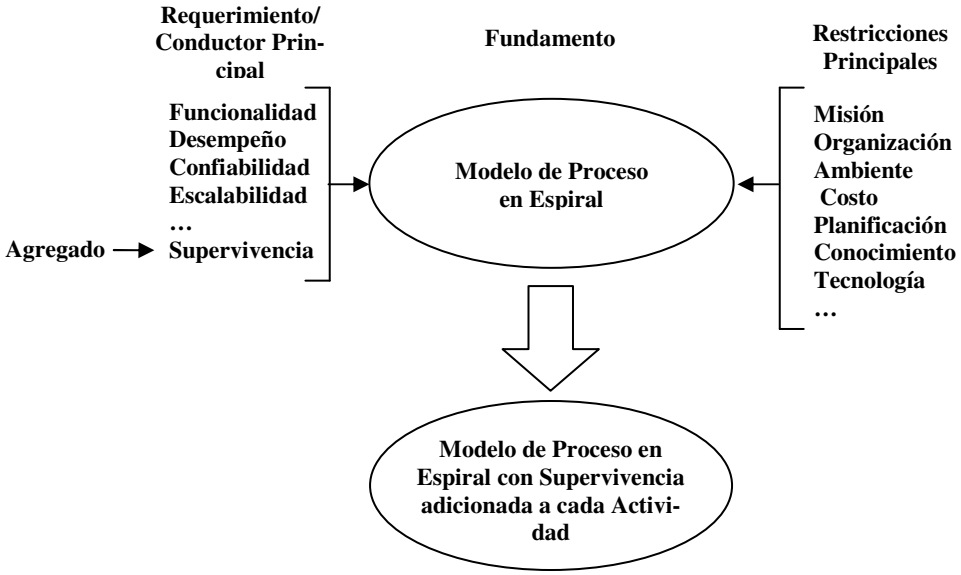


Figura 6. Especialización del modelo en espiral para el conductor supervivencia.

La Tabla 4 identifica *un conjunto de actividades típicas en el desarrollo de un sistema y los correspondientes elementos de supervivencia en cada uno de ellos. El punto clave es que la supervivencia está integrada dentro de actividades más amplias;* por ejemplo, junto con la definición de requerimientos del sistema, deben estar definidos los atributos de supervivencia: función, desempeño, confiabilidad, escalabilidad, y otras propiedades.

Esta tabla identifica *un conjunto de actividades típicas en el desarrollo de un sistema y los correspondientes elementos de supervivencia en cada uno de ellos. El punto clave es que la supervivencia está integrada dentro de actividades más amplias;* por ejemplo, junto con la definición de requerimientos del sistema, deben estar definidos los atributos de supervivencia: función, desempeño, confiabilidad, escalabilidad, y otras propiedades.

Las actividades de la Tabla 4 constituyen el objetivo para la administración del proyecto bajo el modelo en espiral especializado de la Figura 5.

A manera ilustrativa, supongamos la siguiente aplicación del proceso de administración en espiral a la *fase de definición de arquitectura*. Asumimos que las fases anteriores han sido completadas exitosamente y que los documentos apropiados de requerimientos y de especificaciones

están en nuestras manos. La tarea inicial de la espiral de definición de arquitectura inicial es definir un conjunto de componentes candidatos y sus interconexiones que implementarán los servicios especificados de una manera que satisfagan tanto los requerimientos funcionales como no-funcionales. El arquitecto seleccionará las plataformas candidatas, les asignará funciones, y determinará las conexiones apropiadas entre estas plataformas y las del mundo exterior. Se han de utilizar una variedad de herramientas y técnicas para analizar la arquitectura propuesta a los fines de determinar si satisface los requerimientos y especificaciones. Una posibilidad para este

Actividades del ciclo de vida	Elementos claves de supervivencia	Ejemplos
Definición de la misión	Análisis crítico de la misión y las consecuencias de la falla	Estimación del impacto en el costo de los ataques de denegación de servicio
Concepto de operaciones	Definición de las capacidades del sistema en ambientes adversos	Enumeración de las funciones de la misión crítica que deben resistir a los ataques
Planificación del proyecto	Integración de la supervivencia dentro de las actividades del ciclo de vida	Identificación de técnicas de codificación defensiva para la implementación
Definición de requerimientos	Definición de los requerimientos de supervivencia a partir de la perspectiva de la misión	Definición de los requerimientos de acceso para los activos del sistema críticos durante los ataques
Especificación del sistema	Especificación del servicio esencial y los escenarios de intrusión	Definición de los pasos que componen las transacciones del sistema críticas
Arquitectura del sistema	Integración de estrategias de supervivencia dentro de la definición de arquitectura	Creación de una red de facilidades para la replicación de los activos de datos críticos
Diseño del sistema	Desarrollo y verificación de las estrategias de supervivencia	Verificación de la exactitud de los algoritmos de encriptación de datos
Implementación del sistema	Aplicación de codificación e implementación de técnicas de supervivencia	Definición de métodos para evitar vulnerabilidades de *buffer overflow*
Testeo del sistema	Tratamiento de los intrusos como si fueran usuarios durante el testo y la certificación	Agregado de la utilización de intrusión a los modelos de utilización para el testeo estadístico
Evolución del sistema	Mejoramiento de la supervivencia para prevenir la degradación a lo largo del tiempo	Redefinición de la arquitectura en respuesta a los cambios en el ambiente de amenazas

Tabla 4. Actividades de ciclo de vida y los correspondientes elementos de supervivencia.

análisis es que la arquitectura propuesta satisfaga los requerimientos funcionales pero no pueda alcanzar el *throughput* requerido. Si bien la replicación del procesador ya ha sido empleada para la mejora del desempeño, los procesadores requieren de un acoplamiento fuerte para mantener la sincronización, y su ubicación presenta una vulnerabilidad como potencial punto de falla único. Resulta necesaria otra espiral sobre la arquitectura debido a que restan riesgos irresueltos.

Un examen de la especificación del servicio que representaba el cuello de botella demuestra, aquéllo que a primera vista aparece como un servicio monolítico en realidad se descompone de tal manera que reduce la carga de procesamiento y permite la separación del servicio en dos partes, física y temporalmente. Luego de confirmar que la especificación revisada de este servicio satisface los requerimientos y es consistente con las otras especificaciones no modificadas, se revisa la arquitectura. La especificación revisada permite una reducción en la carga del procesador y que la función crítica sea realizada en varias localizaciones distantes con requerimientos fuertemente relajados en lo que hace a la sincronización de datos. Como resultado de ello, resulta posible configurar el sistema con la suficiente redundancia de tal manera que se puedan tolerar al menos dos eventos de pérdida de sitio sin que se produzca la pérdida del servicio. Un mayor número de pérdidas de sitio reducirá los niveles de servicio, pero resulta posible priorizar las solicitudes de tal manera que será mantenido un nivel esencial mínimo. El análisis detallado de esta solución muestra una baja probabilidad de que surja una condición que pudiese provocar un inter-bloqueo del sistema. El agregado de mecanismos de sincronización explícita (otra iteración) y la capacidad de comunicaciones adicionales reduce el riesgo residual a un nivel aceptable, y la fase de arquitectura se completa luego de dos espirales del proceso de administración.

2.3.3 Actividades del ciclo de vida y la capacidad de supervivencia

Los elementos claves de supervivencia de la Tabla 4 son las principales tareas que deben ser administradas dentro del modelo en espiral para alcanzar la supervivencia del sistema. El Método SNA ha demostrado ser de utilidad en las actividades de definición de requerimientos, especificación del sistema, y arquitectura del sistema. Las mismas serán descriptas en este reporte.

2.4 ETAPAS DEL METODO SNA

El Método SNA permite la valoración sistemática de la capacidad de supervivencia de sistemas propuestos, de sistemas existentes y de modificaciones a los sistemas existentes. El análisis se lleva a cabo en el nivel de la arquitectura como un proyecto cooperativo entre un equipo del cliente y un equipo del SEI (Software Engineering Institute). El método se desarrolló vía una serie de sesiones de trabajo conjunto, culminando con una reunión informativa sobre las conclusiones y recomendaciones. La Figura 7 describe el proceso SNA de cuatro etapas.

En la Etapa 1, ***Definición del Sistema***, *se elicita la misión del sistema de negocio y sus principales requerimientos funcionales. Se discute el ambiente de utilización en términos de las capacidades y localizaciones de los usuarios del sistema, y los tipos y volúmenes de transacciones del sistema. Se revisan los riesgos del sistema en términos de los componentes y conexiones de hardware, de las configuraciones de software, y de la residencia de la información.*

En la Etapa 2, ***Definición de las Capacidades Esenciales***, *se seleccionan los servicios y los activos esenciales del sistema*. Los servicios y activos esenciales son aquellas capacidades críticas para satisfacer la misión del sistema de negocio, la cual debe ser mantenida bajo condiciones adversas. Luego *se definen los escenarios de utilización del servicio esencial que invocan los servicios esenciales y el acceso a los activos esenciales*. Los escenarios de utilización están compuestos por los pasos sucesivos requeridos para que los usuarios invoquen los servicios y accedan a los activos. Finalmente, *se trazan los escenarios de utilización a través de la arquitectura del sistema para identificar los componentes esenciales que participan en la provisión de los servicios y activos esenciales*. Este proceso de trazado equivale a la ejecución mental de los escenarios, a medida que los mismos atraviesan los sucesivos componentes dentro de la arquitectura.

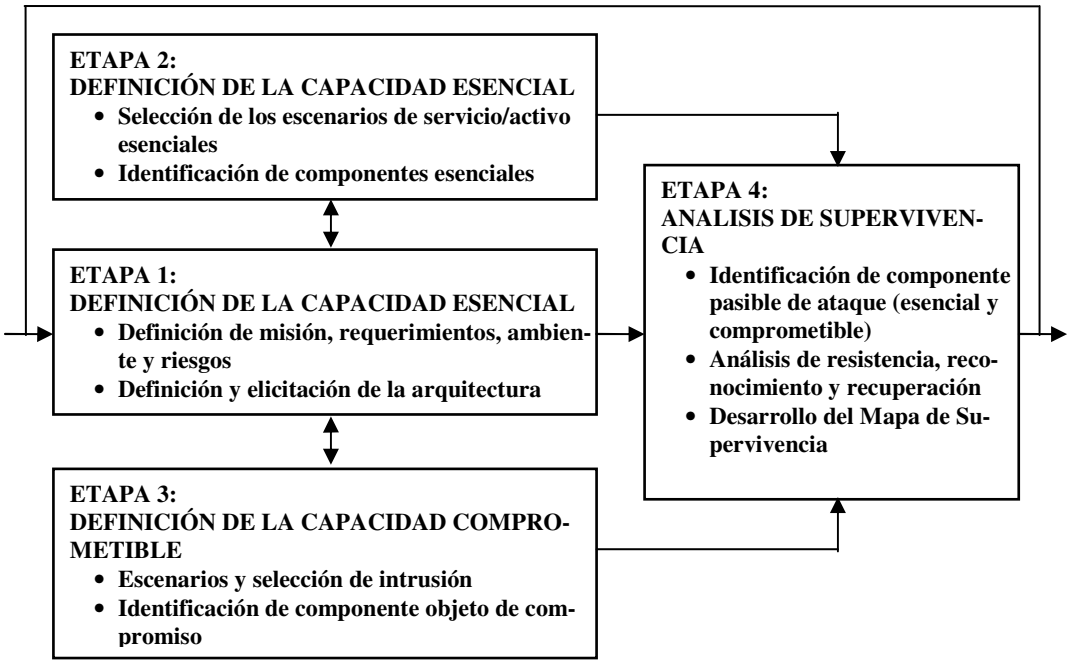


Figura 7. El Método SNA.

En la Etapa 3, ***Definición de las Capacidades Objeto de Compromiso***, *se selecciona un conjunto de intrusiones representativas en base al ambiente operacional del sistema*; *se definen y trazan los escenarios de utilización de intrusión a lo largo de la arquitectura para identificar los componentes objeto de compromiso que las intrusiones podrían acceder con éxito y dañar*.

En la Etapa 4, ***Análisis de la Supervivencia***, s*e identifican los componentes detectados como aquéllos que son tanto esenciales como objeto de compromiso*. Entonces, *se analiza la arquitectura para la protección de estos componentes, en términos de su capacidad de resistir, reconocer y recuperarse frente a intrusiones*. Luego, *se formulan y resumen las recomendaciones respecto de la arquitectura dentro de un Mapa de Supervivencia*, como se ilustra

en la Figura 8 [Ellison 98b].

El *Mapa de Supervivencia relaciona intrusiones y sus correspondientes componentes detectados con estrategias específicas destinadas a mejorar las capacidades del sistema respecto de resistencia, reconocimiento y recuperación.*

Escenario de Intrusión	Efectos softspot	Estrategias de Arquitectura para →	Resistencia	Reconocimiento	Recuperación
(Escenario 1)		Corriente			
…		Recomendado			
(Escenario n)		Corriente			
…		Recomendado			

Figura 8. Plantilla del Mapa de Supervivencia.

Como se indicara, la supervivencia tiene que ver con las condiciones adversas que surgen a partir de las intrusiones, las fallas o los accidentes. El Método SNA se focaliza en las intrusiones y compromisos a los fines de capitalizar tanto la extensa experiencia del CERT/CC en el análisis de intrusión y la base de conocimiento de intrusión del CERT/CC. Es igualmente aplicable al análisis de fallas y accidentes, y este análisis es rápidamente incorporado.

2.5 PROCESO DEL METODO SNA

2.5.1 Planificación y conducción del Método SNA

Un Método SNA se conduce a través de una serie de ***Sesiones Conjuntas*** *–Joint Sessions-* de los equipos del cliente y del SEI, como se describe en el lado izquierdo de la Figura 5. El equipo del SEI también lleva a cabo la serie de ***Tareas Analíticas*** *–Analytical Tasks-* que se muestran en el lado derecho de la Figura 5.

El Método SNA se inicia con una Reunión de Planificación Conjunta *–Joint Planning Meeting-*, y culmina con una Sesión de Resumen Conjunta *–Joint Briefing Session-* que resume los hallazgos y las recomendaciones. Cada etapa en el proceso SNA se describe a continuación. Las Responsabilidades de la Sesiones Conjuntas se denominan "del Cliente", "del SEI" o "Conjuntas" según corresponda.

2.5.2 Reunión de Planificación Conjunta – *Joint Planning Meeting*

Propósito:

Asignar responsabilidades y realizar todos los preparativos para la conducción del Método SNA.

Responsabilidades del SEI:

1. Designar los miembros del equipo del SEI, el que generalmente está compuesto por tres

miembros.

2. Asignar un líder del equipo, quien también servirá como el único punto de contacto (*Point of Contact* – POC).

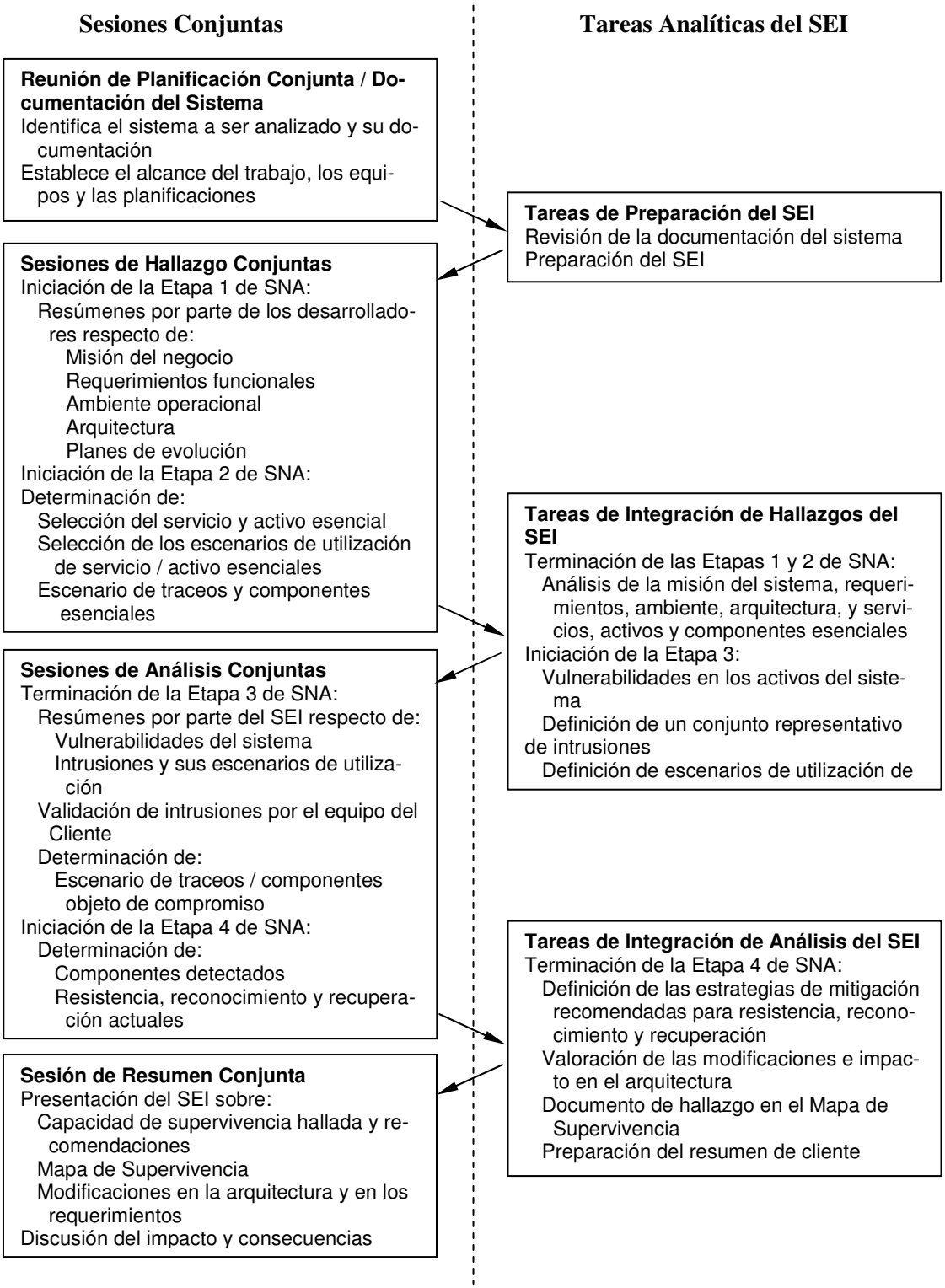


Figura 9. Sesiones y Tareas en el Método SNA.

Responsabilidades del Cliente:

1. Designar los miembros del equipo del Cliente, que debería incluir expertos en la misión, requerimientos, ambiente operacional, utilización y arquitectura del sistema. Por lo general los miembros del equipo del cliente pueden incluir al arquitecto del sistema, un líder de diseño, y varios involucrado, incluido los dueños del sistema (quienes poseen el conocimiento de cómo el sistema atiende a los objetivos del negocio), y los usuarios del sistema (quienes tienen el conocimiento de las necesidades de utilización y patrones). De cuatro a seis miembros generalmente proporcionan una buena representación.
2. Identificar el sistema ha ser analizado. El sistema o parte de él ha ser analizado debería exhibir las siguientes características:
    - Debería tener un tamaño apropiado para permitir el análisis dentro de las restricciones de tiempo y recursos del SNA. Los sistemas grandes pueden ser analizados de manera efectiva con altos niveles de granularidad para producir hallazgos de gran alcance. Los sistemas pequeños se pueden analizar de manera más intensiva para producir hallazgos detallados con un alcance localizado. La granularidad del análisis ser ajustada en función del tamaño y complejidad del sistema.
- Debería exhibir límites claros. Deben ser conocidas y comprendidas todas las interfaces y conexiones del sistema. Por ejemplo, cada conexión de red debería ser conocida

Responsabilidades Conjuntas:

1. Alcance del sistema ha ser analizado y establecer los límites para el Método SNA.
2. Establecer la planificación para el trabajo, y los lugares de reunión para las sesiones conjuntas.

Condiciones de Salida:

1. Están asignados los equipos del SEI y del cliente y sus líderes, está identificado el sistema ha ser analizado, y se ha definido la planificación. La documentación requerida también ha sido identificada, como se describe a continuación.

2.5.3. Documentación del sistema

El Método SNA se ve facilitado por la documentación del sistema objeto de análisis. La documentación generalmente se selecciona de materiales existentes, y proporciona la información aclarativa y de referencia requerida para el análisis. Debe observarse que la documentación puede no estar disponible, resultando de valor documentación parcial en la que los vacíos de información generalmente se cubren durante las sesiones de trabajo.

Responsabilidades del Cliente:

Proveer documentación del sistema al equipo del SEI que describa:

- *Misión del negocio*. Define los objetivos principales del sistema desde una perspectiva del negocio.
- *Requerimientos funcionales*. Definen las principales funciones del sistema en términos de:
    - Principales transacciones disponibles por clase de usuario.
    - Requerimientos sobre acceso y retención de información.
    - Volúmenes y tasas de procesamiento.
- *Ambiente operativo y usuarios*. Define las características operacionales del sistema en términos de:
    - Localizaciones y ambientes de los componentes del sistema.
    - Clases y capacidades de acceso de usuarios, incluidos los usuarios nominales del sistema, desarrolladores, encargados del mantenimiento, operadores y administradores.
    - Procedimientos operativos, incluidos monitoreo y control del sistema, procedimientos de control de usuarios, administración de la seguridad, métodos de mantenimiento, y procedimientos de respaldo y recuperación.
- *Arquitectura*. Define la configuración del sistema en términos de:
    - Componentes de hardware y sus conexiones (generalmente en formato de diagrama de bloques) incluidos los puntos de acceso externos y los enlaces de comunicaciones.
    - Componentes de software residentes en cada componente de hardware, incluidos protocolos, sistemas operativos, aplicativos, bases de datos, repositorios, y facilidades de seguridad, mantenimiento, respaldo y recuperación.
    - Componentes humanos, incluidos los administradores, desarrolladores, encargados del mantenimiento y operadores.

Condiciones de Salida:

Documentación requerida identificada y revisada.

2.5.4 Tareas de preparación

El equipo del SEI revisará la documentación del sistema y preparará las Sesiones de Hallazgo Conjuntas.

2.5.5 Sesiones de Hallazgo Conjuntas – *Joint Discovery Sessions*

Responsabilidades del Cliente:

El equipo del cliente inicia la Etapa 1 del Método SNA, Definición del Sistema, proporcionando resúmenes de la misión del negocio, principales requerimientos funcionales, arquitectura del sistema, ambiente operacional, escenarios de utilización típicos, y planes de evolución del sistema.

Esta información provee la comprensión del sistema a todos los participantes del Método SNA.

Responsabilidades Conjuntas:

Ambos equipos inician la Etapa 2 del Método SNA, Definición de Capacidades Esenciales. El cliente identifica un conjunto de servicios y activos esenciales cuya disponibilidad debe ser mantenida bajo condiciones adversas y los escenarios de utilización que los invocan y acceden. Ambos equipos trazan los escenarios a través de la arquitectura para identificar los componentes esenciales correspondientes. En base a los recursos y las planificaciones disponibles para el Método SNA, generalmente se identifican un conjunto de tres o cuatro servicios y activos esenciales de la más alta prioridad.

Condiciones de Salida:

Ambos equipos comparten un nivel común de comprensión del sistema, se han identificado los servicios y activos esenciales, y se han trazado los escenarios a través de la arquitectura para descubrir los componentes esenciales.

2.5.6 Tareas de Integración de Hallazgos

Responsabilidades del SEI:

El equipo del SEI completa las Etapas 1 y 2 del Método SNA analizando y resumiendo misión, requerimientos funcionales, ambiente operacional, servicios y activos esenciales, trazas de escenario, y componentes esenciales. En base a esta información, el equipo inicia la Etapa 3 del Método SNA, Definición de Capacidades Objeto de Compromiso, valorizando las vulnerabilidades del sistema, y definiendo sus correspondientes escenarios de utilización.

Condiciones de Salida:

Han sido identificadas las vulnerabilidades del sistema y las intrusiones representativas.

2.5.7 Sesiones de Análisis Conjuntas – *Joint Analysis Sessions*

Responsabilidades del SEI:

El equipo del SEI provee un resumen de las vulnerabilidades identificadas y los escenarios de intrusión representativos.

Responsabilidades del Cliente:

El equipo del cliente valida los escenarios de intrusión seleccionados, posiblemente proponiendo modificaciones o extensiones.

Responsabilidades Conjuntas:

Ambos equipos completan la Etapa 3 del Método SNA, trazando los escenarios de intrusión a través de la arquitectura para descubrir los correspondientes componentes objeto de compromiso. Los equipos también inician la Etapa 4 del Método SNA, Análisis de Supervivencia, identificando los componentes destacados (tanto esenciales como objeto de compromiso), y proponen

y discuten potenciales estrategias de resistencia, reconocimiento, y recuperación.

Condiciones de Salida:

Las intrusiones representativas han sido validadas, sus escenarios han sido trazados a través de la arquitectura para descubrir los componentes objeto de compromiso, y se han discutido las estrategias iniciales de mitigación.

2.5.8 Tarea de Integración de Análisis

Responsabilidades del SEI:

El equipo del SEI completa la Etapa 4 revisando los resultados de las Sesiones de Análisis Conjuntas y desarrollando los hallazgos y recomendaciones de mitigación encargadas de las estrategias de resistencia, reconocimiento y recuperación. Las estrategias se definen como modificaciones en la arquitectura actual, las que se resumen en un Mapa de Supervivencia. Se prepara un resumen para el cliente para la revisión de los hallazgos y recomendaciones del Método SNA.

Condiciones de Salida:

Se formulan recomendaciones y se prepara la reseña.

2.5.9 Sesión de Resumen Conjunta – *Joint Briefing Session*

La Sesión de Resumen es conducida por el equipo del SEI encargado de presentar los hallazgos y las recomendaciones desarrolladas durante el Método SNA. Se cubren las siguientes áreas:

- Misión de negocio, requerimientos y ambiente operacional del sistema.
- Arquitectura actual del sistema.
- Servicios y activos esenciales seleccionados y sus escenarios de utilización.
- Componentes esenciales del sistema.
- Intrusiones seleccionadas y sus escenarios de utilización.
- Componentes objeto de compromiso del sistema.
- Análisis de resistencia, reconocimiento y recuperación.
- Modificaciones en la arquitectura recomendadas y Mapa de Supervivencia.

A la Sesión de Resumen concurren el equipo del cliente y su administrador. Los hallazgos y recomendaciones son discutidas y se exploran las próximas acciones.

2.6 RESULTADOS ALCANZADOS

2.6.1 Reporte de cliente del Método SNA

Durante el proceso de ejecución del Método SNA se ha podido determinar que resultan de utilidad un conjunto de diferentes métodos y plantillas orientados al desarrollo de un buen conjunto de recomendaciones. En el Anexo 1 se presenta un Reporte SNA típico.

2.6.2 Lecciones aprendidas

En el trabajo con clientes se ha encontrado cierta variabilidad en el método. Esto puede darse dependiendo de si el cliente está definiendo requerimientos, especificando una arquitectura, o realizando una actualización significativa a un sistema existente. El método SNA se adapta fácilmente para estas situaciones.

Asimismo se ha detectado que ciertos escenarios de ataque le permiten al intruso comprometer todos los activos y servicios, por lo que el mapeo con lo arquitectura se torna trivial y, en consecuencia, todos los componentes esenciales son detectados. Esto conduce a recomendaciones más globales.

2.7 PLANES DE FUTURAS INVESTIGACIONES

El Método *Survivable Network Analysis* ayuda a las organizaciones *a definir e implementar mejoras en el sistema relacionadas con las intrusiones y compromisos inevitables de una manera proactiva. Como implicados que dependen de sistemas para llevar a cabo las misiones de la organización, usuarios, administradores, y personal técnico se han beneficiado de la creciente atención puesta en temas de supervivencia.*

Un próximo paso clave para la evolución del Método SNA es el desarrollo de abstracciones y métodos de razonamiento más potentes para la definición del comportamiento y estructura de sistema distribuidos de gran escala. Tales resultados han de posibilitar un análisis más exhaustivo de los servicios esenciales y de las trazas de intrusión al tiempo de limitar la complejidad.

Además, se requieren mejoras en las representaciones y métodos para la definición de intrusiones. Resulta importante ir más allá de las limitaciones de lenguaje natural, y desarrollar semánticas uniformes para la utilización de intrusión que permitan un análisis más riguroso, e incluso la posibilidad de aplicar métodos computacionales.

Otra línea fructífera de investigación comprende el desarrollo de estilos o plantillas arquitectónicas estandarizadas para las estrategias de supervivencia que puedan ser insertadas y compuestas por arquitecturas de sistema para mejorar sus propiedades de supervivencia. Tales plantillas pueden ser analizadas en forma individual o en conjunto para definir y documentar su contribución a la supervivencia del sistema.

El amplio contexto de la supervivencia, los modelos de ciclo de vida del sistema, y sus actividades asociadas, serán investigadas a los fines de identificar un conjunto estándar de actividades de ciclo de vida que soporte a la supervivencia, y para identificar aquellas actividades del ciclo de vida en las que resulta necesaria una ulterior investigación. En asociación con esto, resultan deseables métricas estandarizadas.

SECCION 3: EL MÉTODO CONTINÚA SU DEFINICIÓN

A finales del año 2002, el equipo formado por R. C. Linger, H. F. Lipson, J. McHugh, N. R. Mead y C. A. Sledge publica el trabajo "*Modelos de Ciclo de Vida para Sistemas con Capacidad de Supervivencia*", en el que se retoma la línea iniciada en los trabajos que fueran reproducidos en las secciones anteriores.

Luego de analizar aspectos ya considerados en "*Survivable Network Method*" –secciones 2.1 a 2.3 inclusive- y "*Survivable Network Systems: An Emerging Discipline*"-sección 1.2-, este nuevo trabajo continúa con la descripción de un modelo de ciclo de vida para el desarrollo de software con capacidad de supervivencia, e ilustra técnicas que se pueden aplicar durante las nuevas actividades de desarrollo dando soporte a los objetivos de supervivencia. Asimismo, describe un modelo de ciclo de vida de software y las actividades asociadas para soportar los objetivos de supervivencia aplicables a sistemas basados en productos COTS.

3.1 ARQUITECTURA Y DISEÑO

En la Figura 13 se describe el nivel arquitectónico de un Método *Survivable Network Design* -SND-. A diferencia de tratar la capacidad de supervivencia como un agregado a posteriori en el diseño y la arquitectura del sistema, *este método integra las consideraciones de supervivencia dentro del proceso de desarrollo*. El mismo se basa en el proceso *Survivable Systems Analysis* [Ellison 98b, Ellison 99a, Mead 00a] desarrollado y aplicado por el CERT/CC.

El Método SND se guía en varios de los tipos de requerimientos-especificaciones descriptos en la Figura 2, específicamente:

- **Requerimientos de sistema y de supervivencia**. Los requerimientos del sistema definen todos los comportamientos funcionales, más las propiedades no-funcionales que un sistema debe satisfacer. *Los requerimientos de supervivencia identifican aquellos elementos del comportamiento funcional que representan servicios esenciales, y elaboran estos servicios en términos de los escenarios de uso de servicios esenciales*.
- **Requerimientos de utilización/intrusión**. Estos requerimientos definen todas las posibles utilizaciones del sistema *bajo circunstancias normales y adversas*. *Los intrusos son tratados con otra clase de usuarios, y se definen escenarios de uso de intrusión representativos*.
- **Requerimientos de operación y de administración con supervivencia**. Estos requerimientos *identifican los procedimiento y políticas que deben ser elaborados*; por ejem-

plo, políticas de seguridad.

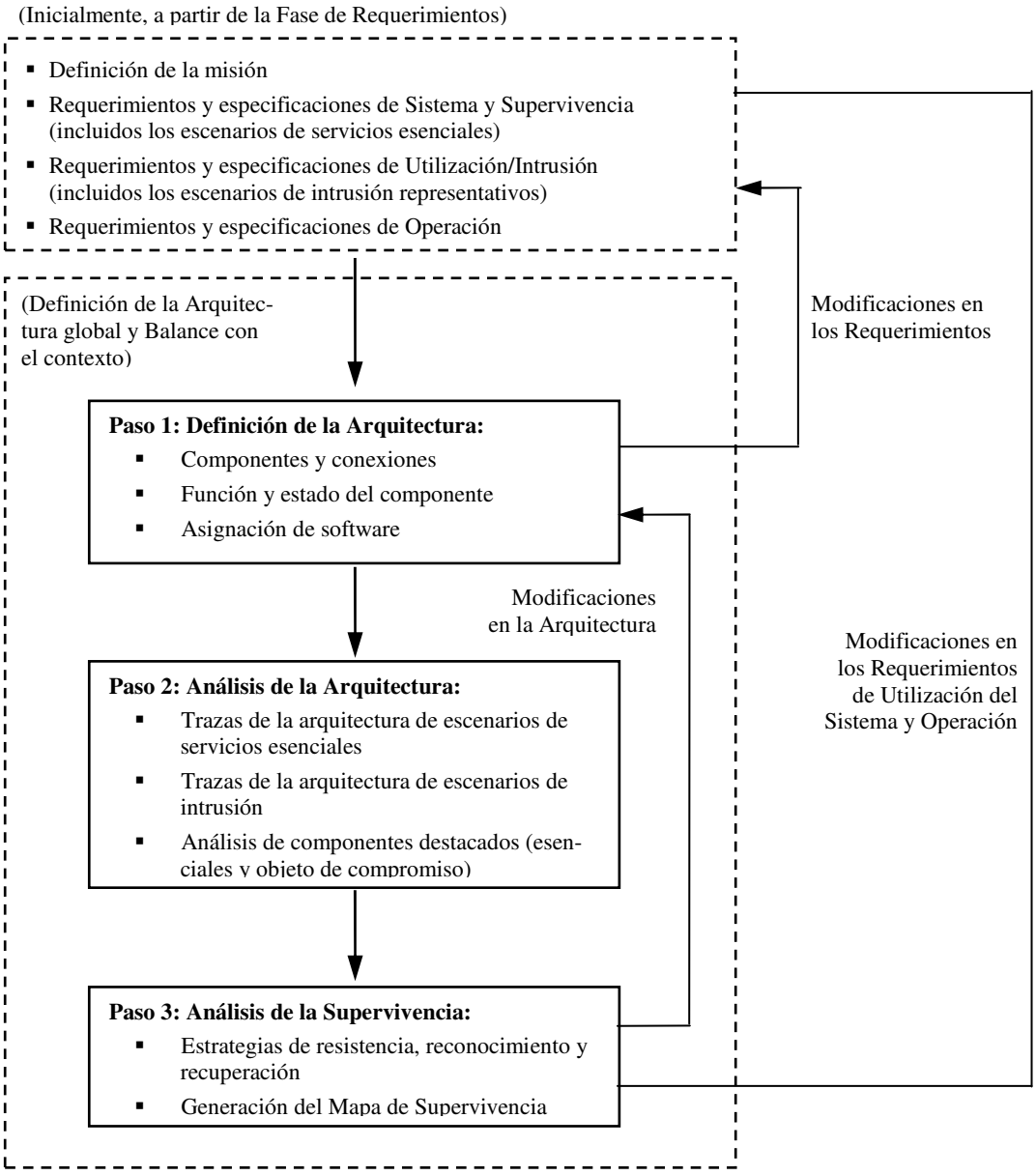


Figura 13. Nivel Arquitectónico de un Método SND

La aplicación arquitectónica de Método SND está embebida dentro de una actividad más amplia de definición y balance arquitectónico que pretende crear una arquitectura del sistema que satisfaga todas las propiedades requeridas, tales como desempeño, aptitud, escalabilidad, costo y mantenibilidad. En la Figura 13, el foco está puesto en el Método SND en la idea de que, en la práctica, este proceso específico de la supervivencia será realizado en paralelo con otras formas de análisis y diseño como, por ejemplo, la simulación a fin de predecir propiedades de desempeño.

La arquitectura del Método SND está compuesta por tres pasos, los que se describen a conti-

nuación:

Paso 1. Definición de la Arquitectura

De acuerdo a la misión y a los requerimientos del sistema se propone una arquitectura. Para ello se seleccionan los estilos y patrones arquitectónicos, y se definen los componentes y conectores. Los componentes se pueden describir en términos de la funcionalidad a ser provista y de los datos de estado a ser conservados. Generalmente, se identifican los elementos específicos de software, incluidos los protocolos, sistemas operativos, ambientes de ejecución, y aplicaciones.

Paso 2. Análisis de la Arquitectura

Se investigan las propiedades de supervivencia de la arquitectura candidata en términos de (1) servicios esenciales (servicios que deben ser mantenidos durante el ataque); (2) activos esenciales (activos cuya integridad, confidencialidad, disponibilidad, y otras propiedades deben ser mantenidos durante el ataque); y (3) objetivos de la misión y las repercusiones de la falla. En primer lugar, se caracterizan los usos del servicio y los activos esenciales mediante *escenarios de utilización*, los que se identifican dentro de la arquitectura como *trazas de ejecución* para identificar el conjunto de *componentes esenciales* asociados (componentes que deben ser capaces de entregar los servicios esenciales y de mantener los activos esenciales). Luego, se seleccionan escenarios de intrusión representativos en base al ambiente del sistema y a la evaluación de riesgos y aptitudes del intruso. Las selecciones también están influenciadas por la vasta base de conocimiento de estrategias de intrusión que posee el CERT. Estos escenarios son de igual manera trazados sobre la arquitectura como trazas de ejecución para identificar los conjuntos de *componentes objeto de compromiso* (componentes que podrían se penetrados y dañados por la intrusión). En este trazado, el método SND toma en cuenta fortalezas y debilidades de los componentes COTS, como así también cualquier defecto de confiabilidad y de seguridad conocidos. Finalmente, se identifican los *componentes destacados* de la arquitectura, como aquellos componentes que son tanto esenciales como pasibles de ser comprometidos, en base a los resultados previos.

Paso 3. Análisis de la Supervivencia

Luego *se analizan los componentes destacados y sus arquitecturas de soporte* en lo que hace a sus propiedades claves para la supervivencia: resistencia, reconocimientos y recuperación. En este punto, se evalúan las estrategias de supervivencia y los patrones arquitectónicos en pos de usos potenciales en el mejoramiento de la supervivencia de la arquitectura candidata. Este análisis de las "tres Rs" se resume en un *Mapa de Supervivencia*. Este mapa es una matriz que enumera, para cada escenario de intrusión y sus correspondientes efectos sobre componentes destacados asociados, las *estrategias arquitectónicas* actuales y recomendadas en lo que hace a resistencia, reconocimientos y recuperación. *El Mapa de Supervivencia proporciona una retro-*

alimentación sobre la arquitectura original y, a menudo, da por resultado un proceso interactivo de análisis costo/beneficio y mejoramiento de la supervivencia. El mismo también puede proveer retro-alimentación respecto de los requerimientos del sistema, como una nueva comprensión y mejores ideas emergentes del análisis.

La solución basada en escenarios del Método SND es una generalización de los métodos *operation-sequence* [Kemmerer 91] y métodos de escenarios de utilización [Carrol 99, Prowel 99].

3.2 IMPLEMENTACIÓN Y VERIFICACIÓN

3.2.1 Estrategias de Codificación Defensiva[7]

Muchas vulnerabilidades de intrusión resultan ser el resultado de prácticas pobres de codificación que provocan comportamientos no intencionados del programa pero susceptibles de abuso, que los intrusos pueden emplear para ganar acceso. Por tal motivo, *la gestión del desarrollo de sistemas con supervivencia requiere de la definición y efectiva aplicación de estándares y prácticas de codificación defensiva*.

Las especificaciones de sistema tratan con comportamientos de alto nivel establecidos bajo la suposición de que los programas operan dentro de las restricciones impuestas por la semántica tanto del lenguaje de especificación como del lenguaje de programación o implementación. Y, por lo general, las mismas asumen que esta operación se desenvuelve dentro del mundo matemático de los números reales y los enteros arbitrariamente grandes. Sin embargo, los lenguajes de programación operan con números de coma flotante y enteros dentro de un rango restringido estricto. Debido a que sería deseable poder demostrar que el comportamiento del programa no es inconsistente con el comportamiento especificado, se requiere de una especial atención para asegurar que las operaciones del programa producen valores que se mantienen dentro de estos rangos y, al menos en el caso de los enteros, son idénticos a los resultados matemáticamente esperados.

Los lenguajes de programación varían en su tratamiento de los programas que violan estas restricciones. En general, se debería prestar especial atención a la clase de restricciones que especifican los valores legítimos que puede asumir una variable de un tipo de dato dado. Estas, a su vez, limitan las operaciones que se pueden realizar con tales variables. Por ejemplo, el resultado de realizar una división entera con un divisor cero no está definido y el resultado de tal división no se puede asignar, en general, a una variable entera. De manera similar, las operaciones aritméticas que provocan desbordamientos (*overflows*) de hardware no son consistentes con las definiciones matemáticas de las mismas operaciones.

[7] Las técnicas de codificación o programación defensivas son una forma de diseño defensivo orientado a asegurar la continuidad de la función de una porción de software pese al uso imprevisible de dicho software.

Lenguajes como Ada proveen mecanismos de excepción que detectan intentos de violación de suposiciones del lenguaje en tiempo de ejecución. Los programadores pueden elegir proporcionar código que se ocupe de estas condiciones excepcionales y intente su recuperación o permita que el *run-time* del sistema termine el programa cuando se viole una restricción. Lenguajes como C dejan indefinido el comportamiento del programa frente a excepciones (o definido por la implementación) y ofrecen poca o ninguna ayuda al programador en lo que hace a evitar las excepciones o a hacerles frente cuando ellas ocurren; el efecto de un desbordamiento en un programa C por lo general es la asignación a una variable de un resultado legal, pero incorrecto, y el subsiguiente uso de tal resultado puede conducir a una cascada de resultados erróneos que culminan con la entrega de resultados erróneos a un usuario o en un problema grave que el programa finaliza de manera anormal, probablemente debido a un error de direccionamiento a nivel de hardware.

Cuando se accede a un *array*, el descriptor utilizado debe referenciar a un elemento dentro de los límites declarados de dicho arreglo. De manera similar, cuando se manipulan los arreglos como un todo (por ejemplo, se concatenan *strings*), se asume que el destino de la operación es lo suficientemente grande como para contener al resulto completo. Dado que, por lo general, los arreglos están asociados a bloques contiguos del espacio de direcciones, la lectura desde una dirección que no es parte del arreglo tiene uno de dos efectos: (1) la dirección no parte del especia de direcciones del programa y ocurre algún tipo de fallo, o (2) la dirección es asignada a alguna otra entidad - código, otra variable, o una memoria no utilizada pero direccionable (*slack space*). En el primer caso, generalmente se produce una terminación anormal del programa. En el segundo caso, los resultados son impredecibles a menos que el código, los datos o el *slack space* asociado sean conocidos. Si la operación con el arreglo es un *store*, el resultado podría ser un cambio inesperado en el valor de otra variable o la corrupción de código de programa. Si se conoce de qué manera está asignado el código y los datos por el compilador y el cargador, es posible que las operaciones por fuera de los límites del arreglo modifiquen código y logren cambios predecibles en el comportamiento del programa. Comenzando con el gusano Morris en el año 1988, éste se ha vuelto un método común en los programas que desencadenan ataques vía Internet.

La mayor parte de las violaciones, tanto las del tipo *escalar* como del tipo *array*, son casi siempre evitables. Los lenguajes con semánticas de excepción adecuadas como Ada, proveen indicaciones en *run-time* de los intentos de acceso a elementos del *array* por fuera de los límites de éste. Los lenguajes *type-safe*[8] como Java, proporcionan protecciones similares. En C, los programadores pueden codificar chequeos explícitos para los *overflows* y los accesos *out-of-bounds*. Esto no siempre se hace por diversas razones. Generalmente se piensa que tales che-

[8] *Type-safe* es uno de los fundamentos del modelo de seguridad de Java. Desafortunadamente, los *errors* existentes en el *type checking system* es una de las manera de en que pueden ser atacados los programas Java y las JVMs.

queos demandan un costo inaceptable en el *run-time*. Asimismo, la mayoría de los programadores simplemente no piensan sobre las formas en que su programa podría ser forzado a fallar mediante en ingreso de *inputs* anormales. Las técnicas de programación defensiva pueden evitar un amplio especto de clases de fallos de programa que resultan de *inputs* anormales. Si se las aplica cuidadosamente, estas técnicas no necesitan de una significativa penalización en el *run-time*. En general, la especificación del programa debería describir los rangos de entrada sobre los que se espera que el programa trabaje.

El primer paso en la construcción de un sistema defensivo es la de agregar código para chequear que los datos se encuentran dentro del rango esperado. En el caso de *arrays* o estructuras, esto significa leer los datos en grupos de tamaño definido. El gusano Morris original sacaba provecho del hecho que el programa alterado asumía que sus líneas de entrada debían ser del tamaño correcto (92 bytes) y utilizaba una rutina *gets* hasta encontrar un carácter *newline*, colocando los caracteres leídos dentro de un *array*. Una versión defensiva podría ser contar los caracteres a medida que se los lee, descartando las entradas que fueran muy cortas o muy largas y luego asegurándose que las entradas de longitud correcta tienen la estructura apropiada.

Una vez que las entradas han sido validadas, por lo general resulta posible razonar acerca de excepciones adicionales. Si las longitudes de los *strings* que están siendo concatenados han sido previamente chequeadas, los mismos pueden ser combinados de manera segura si el destino de la concatenación es lo suficientemente grande como para mantener sus longitudes máximas combinadas. Se aplican razonamientos similares a las excepciones que podrían surgir a partir de las operaciones aritméticas. La experiencia con una diversidad de programas muestra que, además de los chequeos de entrada (lo cual siempre debería llevarse a cabo dado que las entradas no están bajo el control del programador), generalmente se requieren unos pocos chequeos adicionales. La optimización de los compiladores para los lenguajes *type-safe* realizan tales razonamientos a los fines de eliminar chequeos en el *run-time* [McHugh 84]. Se requiere del uso de estas técnicas y otras relacionadas en sistemas que se pretende sean evaluados para niveles superiores de aseguramiento del *Trusted Computer Security Evaluation Criteria* [DoD 85], y Young y McHugh [Young 87] los analizan en detalle.

La inspección del diseño y la codificación son métodos excelentes para el aseguramiento de una aplicación uniforme de las prácticas defensivas. Las inspecciones pueden estar integradas dentro de la espiral del proceso de administración como un medio para poner de manifiesto y mejorar el desempeño del desarrollo y la adhesión a los estándares del proyecto.

3.2.2 Verificación de la Exactitud

La mayoría de las vulnerabilidades de intrusión son el resultado de prácticas pobres de progra-

mación que producen un comportamiento imprevisto del software que, a menudo, resulta de utilidad a propósito de una intrusión –por ejemplo, el no planeado y ampliamente abusado comportamiento asociado con los problemas de *buffer-overflow*. *La primera y mejor línea de defensa contra la intrusión es el software cuyo comportamiento requerido, bajo todas las circunstancias posibles de uso, está completamente especificado, y cuyo comportamiento de implementación ha sido verificado contra estas especificaciones.*

Debido a que los sistemas que pueden ser atacados en todos los niveles deben ser defendidos en todos los niveles, resulta importante verificar todos los componentes de software sobre su corrección con respecto a las especificaciones. Por su propia naturaleza, el testeo resulta insuficiente para esta finalidad. Aún los sistemas que comprenden componentes pequeños de software exhiben una población casi infinita de posibles ejecuciones; aún los programas de testeo más cuidadosamente concebidos se pueden ejercitar no más de una fracción de minuto de estas ejecuciones. En realidad, todo el testeo es un muestreo de una población esencialmente infinita de posibles ejecuciones. Por otro lado, *la verificación de la corrección tiene la intención de examinar el comportamiento funcional global del software, y no está limitado a los procesos de ejecución particulares*.

La verificación *function-theoretic*[9] se adecúa particularmente muy bien para este propósito [Linger 99, Prowell 99]. Permite que el equipo de desarrollo verifique de manera completa la corrección del software respecto de las especificaciones. Un *Teorema de Corrección* (*Correctness Theorem*) define las condiciones que se deben cumplir para alcanzar un software correcto. Estas condiciones se verifican a través de patrones de razonamiento de corrección sistemáticos y repetibles aplicados durante revisiones especiales de equipo. En tanto los programas contienen un número esencialmente infinito de caminos, los mismos están compuestos de un número finito de estructuras de control anidadas y secuenciadas (*sequense, ifthenelse, whiledo*, etc.). El Teorema de Corrección se basa en la verificación de estructuras de control, una tarea finita, y no en la traza de sus caminos de ejecución, una tarea abierta. Esta reducción de la verificación a un proceso finito permite el chequeo de la corrección de toda la lógica del software, para ayudar a asegurar que sean eliminados de los diseños comportamientos imprevistos y vulnerabilidades de potenciales intrusiones.

Las condiciones de corrección definidas por el Teorema de Corrección para las estructuras de control fundamentales se indican en la Tabla 5. Las estructuras de control se encuentran expresadas en un lenguaje de diseño de formato genérico. Sobre la izquierda, cada estructura de control está precedida de una definición de función etiquetada *f* que define y documenta su efecto

[9] La visión *function-theoretic* de los programas propone la posibilidad de un cálculo automatizado del comportamiento de un programa. Si bien existen muchos desafíos teóricos, el valor del cálculo del comportamiento de sistemas de alto-aseguramiento podría ser sustancial.

neto sobre los datos –esto es, la asociación especificada desde el dominio al rango que la estructura de control ha de implementar. Dentro de las estructuras, *g* y *h* representan operaciones sobre los datos. Sobre la derecha, la secuencia de consultas de corrección comprende la función de composición, el *ifthenelse*, el caso de análisis, y el *whiledo*, un argumentos de terminación más el caso de análisis y la función de composición combinados. Prowell y Stavely [Prowell 99, Stavely 99] proporcionan una explicación completa de la verificación *function-theoretic*.

Estructura de control	Consulta de corrección
Sequense: *[f]* *do* *g;* *h* *enddo*	¿*g* está seguido por *h do f?*
Ifthenelse: *[f]* *if* *p* *then* *g* *else* *h* *endif*	Cuando *p* es *true*, ¿es *g do f*?, y cuando *p* es *false*, ¿es *h do f*?
whiledo: *[f]* *while* *p* *do* *g* *enddo*	¿El bucle termina, y cuando *p* es *true*, *g* está seguido por *f do f*; y cuando *p* es *false* no se hace *f*

Tabla 5. Condiciones de corrección para la verificación funcional.

3.3 TESTEO

En la administración del desarrollo de los sistemas con capacidad de supervivencia, resulta importante tratar al testeo de la supervivencia como equivalente al testeo de la funcionalidad, el desempeño y otros atributos del sistema. El ***testeo de penetración*** *y el* ***testeo estadístico basado en la utilización*** *son dos métodos útiles para la evaluación de la supervivencia del sistema.*

3.3.1 Testeo de Penetración

A menudo llamados "*red teams*", los que participan de un testeo de penetración, intentan comprometer un sistema de una manera benigna, y así evaluar la efectividad de las defensas del sistema contra un ciber-ataque. *El testeo de penetración ofrece un método complementario de evaluación de la seguridad de un sistema, pero nunca sustituye al tradicional testeo o certificación del sistema*. Se lleva a cabo con el permiso de la organización propietaria del sistema, y

dentro de los límites de las normas básicas que especifican qué queda fuera de dichos límites y qué no. Para alcanzar una máxima efectividad, el equipo de testeo debería ser libre de utilizar una vasta variedad de técnicas de recolección de información, incluidas herramientas de sondeo, ingeniería social, y *dumpster diving*[10], para apoyo de sus subsiguientes ataques benignos sobre el sistema. Esto le permite al equipo testear la seguridad de la organización en su conjunto, de la cual el sistema es sólo una parte.

La incorporación del concepto de supervivencia en el testeo de penetración puede aumentar su efectividad y valor. Los conceptos de seguridad computacional le proporcionan al equipo de testeo de una primera idea acerca de qué es lo que está peor dentro de un sistema. La supervivencia le proporciona una guía mucho más fuerte, porque sólo si los servicios críticos para la misión han sido interrumpidos, han resultado exitosos los esfuerzos del equipo de testeo. El ataque con éxito de partes no-esenciales de un sistema no le permite al equipo declarar victoria. Este uso "estratégico" del testeo de penetración puede estar ligado a un ciclo de vida del sistema y al diseño evolutivo de los sistemas con supervivencia. A medida que se modifica la misión del sistema (es decir, evoluciona a lo largo del tiempo), los servicios esenciales que soportan la misión pueden cambiar, y en consecuencia, variar los objetivos que un equipo de testeo debe atacar exitosamente para vencer las estrategias de supervivencia del sistema.

3.3.2 Testeo basado en Estadísticas de Utilización

Como se indicara anteriormente, cualquier proceso de testeo puede ejecutar sólo un pequeña muestra de las posibles ejecuciones del sistemas. *El problema y la oportunidad en el testeo es de qué manera diseñar la muestra. Queda en evidencia que si la muestra es representativa del eventual campo de utilización, los resultados del testeo pueden proporcionar predicciones científicamente válidas del campo de experiencia con el software*. En esta aproximación, *el testeo es conducido como un experimento estadístico, y los resultados se pueden utilizar para predecir en términos estadísticos de qué manera se comportará el software bajo todas las ejecuciones no testeadas. Esta forma del testeo permite una certificación de la aptitud del software para un determinado uso*, y se encuentra descripta de manera completa en el libro de Prowell y otros [Prowell 99].

En términos generales, le proceso comienza con la *construcción de un modelo de utilización que enumera los posibles usos del software y sus probabilidades de ocurrencia*. A menudo, los modelos de uso se expresan en términos de gramáticas formales o de cadenas de Markov. Luego, se pueden *definir las muestras del modelo conforme a las probabilidades, para identificar un con-*

[10] Persona que buscan cosas que otras personas han tirado y que todavía son útiles, pueden ser recicladas, y poseen valor. Técnica que se utiliza para la búsqueda de información que no se podría utilizar de otra forma, las organizaciones y las personas frecuentemente descartan información crítica (listados de contraseñas, números de tarjeta de crédito, planes de negocio, etc.), la que puede ser recuperada.

junto de casos de testeo fiel a la distribución de probabilidades definida. Finalmente, *estos casos se pueden ejecutar, y sus resultados (éxito o falla) usados para predecir un eventual campo de experiencia con el software*.

En el contexto de la supervivencia, los intrusos son tratados como otra clase de usuarios del sistema, y la utilización de intruso se integra dentro de un modelo de utilización con la utilización legítima. Cuando se ha muestreado el modelo, la utilización de intrusión aparecerá dentro de los casos de testeo junto con la utilización legítima. El éxito o la falla de los usos de intrusión se pueden utilizar para evaluar y predecir las propiedades de supervivencia en el campo de uso.

3.4 EVOLUCIÓN DEL SISTEMA

El diseño evolutivo es un concepto importante que impregna el ciclo de vida de todos los sistemas de información complejos, pero la evolución juega un rol particularmente crucial en el ciclo de vida de los sistemas con capacidad de supervivencia. Esto se debe a que el principal foco de la supervivencia de la información está puesto en la protección de la misión del sistema contra acciones maliciosas de adversarios inteligentes. *Las capacidades de los adversarios inteligentes no son estáticas, sino que evolucionan a lo largo del tiempo en fortaleza y penetrabilidad*. La sofisticación de las técnicas de ataque está evolucionando constantemente. Tanto el conocimiento de estas técnicas como el soporte automatizado en la forma de *scripts* y herramientas de ataque fácilmente disponibles, se están difundiendo de manera continua a través de Internet. Más aún, el CERT/CC y otros equipos similares de respuesta a incidentes están encontrando evidencias de una mejora en las técnicas de ingeniería de software empleadas en el diseño de algunos de los más recientes *scripts* de ataque. Todo esto se traduce en una carrera siempre creciente entre atacantes y defensores, que continuará en tanto existan los sistemas de software basados en red.

La supervivencia es una disciplina fundamental que combina la seguridad computacional con la administración de los riesgos del negocio [Lipson 99]. La constante evolución del sistema, basada en la continua valoración del riesgo a lo largo del ciclo de vida del sistema, es esencial para el diseño de sistemas con supervivencia. En un ambiente típico de mantenimiento, la visión original de la arquitectura no se preserva y la integridad del sistema se degrada a lo largo del tiempo. *En ausencia de una constante evolución del diseño del sistema administrada por el riesgo, la seguridad y la supervivencia del sistema también se degradarán*. Por ejemplo, continuamente se están descubriendo nuevas vulnerabilidades en muchos sistemas que incluyen componentes COTS, y las configuraciones del sistema quedan a la deriva respecto de sus configuraciones óptimas. Los requerimientos de misión y supervivencia pueden cambiar y ya no estar

más reflejados en el diseño existente del sistema. Finalmente, como se indicara anteriormente, las técnicas de ataque están evolucionando de manera continua y pueden exceder la capacidad del sistema de una adaptación automática.

Es necesario distinguir el diseño evolutivo de los sistemas con supervivencia de las simples actividades de adaptación y mantenimiento (posiblemente automatizadas), tales como una actualización de las definiciones de virus, el agregado de nuevas reglas y patrones de ataque a una facilidad de del sistema de detección de intrusión, el *tunning* de un *firewall* o los avisos de monitoreo de la seguridad y los parches que anuncias vulnerabilidades de seguridad en componentes COTS. ***Por el otro lado, las actividades de mantenimiento complejas, las cuales pueden incluir capacidades nuevas o mejoradas, deberían ser consideradas parte de un diseño evolutivo***. La exitosa aplicación del diseño evolutivo en los sistemas con supervivencia depende del establecimiento de una actividad de "vigilancia de la supervivencia", la cual comprende el continuo monitoreo del sistema y su ambiente para detectar cambios que puedan afectar las suposiciones realizadas en relación con la administración del riesgo sobre las que se funda la supervivencia del sistema. Esto habla fuertemente a favor de *la formación de un equipo de valorización del riesgo de la supervivencia* (*Survivability Risk-assessment Team* – SRT), *el que debería ser responsable de la actividad de vigilancia de la supervivencia dentro del equipo de diseño del sistema. Los recursos dedicados al SRT y a la vigilancia de la supervivencia dependerán de la tolerancia al riesgo de la gestión ejecutiva y de su percepción de la relación costo-beneficio de esta actividad.*

La valorización del riesgo para la supervivencia requiere de un amplio rango de perspectivas y destrezas, por lo que los miembros del SRT deben provenir de todos los niveles de una organización (gestión ejecutiva, expertos en el dominio de aplicación, expertos en seguridad computacional, y otros participantes, incluidos los clientes). Los SRTs para sectores particulares de la industria y el gobierno se pueden formar para proveer algún tipo de asistencia genérica a los SRTs organizacionales, pero la naturaleza de misión crítica de la supervivencia significa que los SRTs en el nivel organizacional deben tener la responsabilidad última de la valorización del riesgo para al supervivencia.

Se utiliza el término "*triggers de valorización del riesgo*" para hacer referencia a *los elementos de un sistema o su ambiente que debería monitorear un SRT, buscando los cambios que pueden afectar las suposiciones de valorización del riesgo que son la base de la supervivencia de un sistema. Le corresponde al SRT determinar si un cambio particular o un conjunto de cambios dispararán una actividad de diseño evolutivo y decidir acerca de la extensión de esa actividad.* La Tabla 6 contiene un conjunto representativo de elementos disparadores a los que un SRT debería ir siguiéndole la pista en relación con sus cambios. Los eventos disparadores incluyen cambios en las técnicas de ataque, en la misión, en la gestión, en el *staff*, en los clientes y en los

ambientes tecnológicos y legales.

Un cambio en uno o más elementos disparadores puede iniciar alguna de las variadas actividades de diseño evolutivo descriptas en la Tabla 7, variando desde ninguna acción, la realización de una o más actividades de ciclo de vida de supervivencia, hasta el abandono de un sistema. *El SRT debería iniciar las actividades de diseño, pero el equipo de diseño del sistema debería ser el responsable de llevarlas a cabo*.

Por ejemplo, un experto en seguridad computacional del SRT toma conocimiento de una nueva técnica de ataque que podría amenazar a la supervivencia del sistema existente. Supone que esta nueva técnica de ataque no puede ser contrarrestada por las actividades de mantenimiento simples tales como la aplicación de un parche de seguridad a un componente del sistema o el agregado de una nueva regla del *firewall*. En base a esta nueva técnica de ataque, el experto de seguridad genera un conjunto de nuevos escenarios de ataque para que sean utilizados como una entrada delta a un *Survivable System Analysis* -SSA- del sistema existente. Si se descubren deficiencias en la resistencia, reconocimiento o recuperación del sistema, entonces se ha descubierto una o más actividades del ciclo de vida de la supervivencia (tal como una modificación de la arquitectura del sistema, o un cambio en los requerimientos de la supervivencia).

La terminación de una actividad del ciclo de vida de la supervivencia puede disparar la necesidad de otro. También puede resultar necesario realizar ajustes en el balanceo de diseño con otros atributos de calidad del sistema. El punto en el cual el proceso de diseño evolutivo se detiene depende de la tolerancia al riesgo de una organización, y la relación costo-beneficio percibido con respecto a un conjunto particular de eventos disparadores. Si la evolución no es factible, la organización puede tolerar el riesgo o buscan otras alternativas que transcienden al sistema.

Resulta imperativo que las actividades de diseño evolutivo tengo lugar dentro del contexto de un acceso completo al conjunto total de artefactos del proceso de diseño (tales como descripciones de la racionalidad de los balanceos llevados a cabo durante el último ciclo de vida). *La continuidad del equipo de diseño es particularmente crucial para el diseño evolutivo de los sistemas con supervivencia, de tal manera que la experiencia del diseño de la misión específica pueda ser sostenida a lo largo de la vida del sistema. De lo contrario, el proceso de diseño evolutivo puede degenerar en una aproximación de "emparchado" que nunca puede soportar la supervivencia a largo plazo de los sistemas*. Así como la supervivencia debe estar diseñada dentro de un sistema desde el comienzo y no agregada como algo tardío, la supervivencia a largo plazo no puede ser sustituida mediante el mantenimiento de rutina y la aplicación de parches, sino que sólo a través de la incorporación continua de nuevas soluciones de supervivencia mediante un proceso de diseño evolutivo basado en principios.

Elementos disparadores de una Actividad de Diseño Evolutivo	Ejemplos
Técnicas de ataque	Ha sido descubierta una nueva técnica de ataque, o la variación de una existente, para la que el sistema no puede adaptarse de manera automática o no puede ser protegido mediante una rutina de mantenimiento (por ejemplo, simplemente agregando un nueva regla para la resistencia, reconocimiento o recuperación)
Misión, servicios esenciales, atributos de calidad esenciales, recursos y activos de información clave	La misión de la organización ha cambiado o el sistema ha de ser comprado o implantado por otras organizaciones con misiones diferentes.
Clientes	Clientes nuevos que pueden resultar menos conocidos (y por ende en los que se deposita menos confianza), pueden requerir de un acceso más extensivo a los recursos o activos de información, o pueden requerir una calidad de servicio superior (por ejemplo, aumento en la disponibilidad) que la que requieren los clientes anteriores.
Gestión	La nueva gestión ejecutiva puede diferir en su tolerancia al riesgo y sus estrategias de administración del riesgo.
Workflow y procesos	Cambios en los procesos organizacionales a los que contribuye el sistema pueden afectar la supervivencia global de la misión. Estos pueden ser nuevas formas de atacar al sistema o la interfase humano-máquina.
Staff organizacional	El volumen de ventas puede provocar una reducción del equipo de expertos, lo que puede estresar la supervivencia del sistema. Con un rápido crecimiento de la organización, el nuevo equipo puede ser objeto de una menor confianza que el anterior (por ejemplo, puede faltar tiempo para chequear los antecedentes y los empleados pueden estar emplazados en sitios distantes).
Fabricantes	Un nuevo fabricante de un componente del sistema puede requerir mantenimiento remoto y acceso de confianza.
Colaboradores	Un *partner* de un proyecto puede ser un competidor en el siguiente.
Dependencias e interdependencias	El incremento en la dependencia de un sistema puede estar provocado por la eliminación de los procesos manuales, las posiciones del *staff*, o sistemas *legacy*, lo cual significa que ya no existe una alternativa en caso que el sistema falle. Otro ejemplo es el constante incremento de la interdependencia entre las infraestructuras críticas del país.
Tecnología	Un cambio en el ambiente tecnológico en el cual opera el sistema puede reducir la efectividad de una estrategia de seguridad o de supervivencia. (Esto incluye cambios en el ambiente ilimitado de los sistemas, nuevas técnicas de seguridad, y cambios en la disponibilidad y el conocimiento de la tecnología en el dominio de aplicación).
Ambiente de amenaza	Competidores industriales más agresivos o aumento en el ciber-terrorismo respaldado por países pueden requerir que se destinen a la supervivencia recursos de sistema adicionales.

Componentes del sistema	Un componente COTS que ya no está más soportado puede haber sido reemplazado con uno nuevo cuya contribución, positiva o negativa, al sistema puede haber evolucionado.
Utilización, funcionalidad, acceso a calidad de servicio	Nuevos medios de acceso a un sistema (por ejemplo, inalámbrico), nuevas formas de utilizar un sistema existente, o nuevas expectativas en cuanto a la calidad de servicio pueden afectar la supervivencia de un sistema.
Costo, ganancia y otros factores económicos	El cambio de factores económicos puede amenazar o favorecer la supervivencia de un sistema al afectar la relación costo-beneficio asociada con las diferentes soluciones de supervivencia (por ejemplo, estrategias de mitigación del riesgo). La viabilidad es un factor principal que se debe balancear con la supervivencia. La disminución en el costo de componentes puede conducir a un rediseño evolutivo que provoca redundancia y diversidad mejoradas para soportar una capacidad de supervivencia. El aumento en las demandas de los participantes por beneficios de corto plazo puede inclinar los requerimientos de supervivencia hacia un riesgo mayor.
Marco legal	El uso de un sistema en una nueva jurisdicción más estricta puede incrementar los riesgos de obligaciones y amenazas sobre la supervivencia. Nuevas leyes, el aumento en la obligatoriedad de las leyes existentes, o acciones legales pueden cambiar la ecuación riesgo y amenazar la misión.
Regulaciones gubernamentales	Cambios en regulaciones gubernamentales en apoyo de un aumento en la privacidad, protección, competencia o calidad el servicio pueden disparar la necesidad de modificar el diseño de un sistema a fin de asegurar la continuidad de su supervivencia. (Por ejemplo, la desregulación del sistema de energía eléctrica en los EEUU aumentó la competencia, pero redujo la fiabilidad).
Requerimientos de certificación o estándares	La interrupción del seguro del negocio con cobertura sobre los ciber-ataques puede depender de la certificación de la supervivencia del sistema o de la demostración de que el mismo satisface un estándar mínimo. En consecuencia, estándares nuevos o cambios en los existentes o requerimientos de certificación pueden afectar la supervivencia.
Entorno político y social	Los cambios en lo que hace a privacidad, relaciones de confianza, o la tolerancia al riesgo de la sociedad en su conjunto afectarán los requerimientos de supervivencia de las infraestructuras nacionales críticas de las que depende la sociedad.
Experiencia operacional (ataques, accidentes y fallas)	La retroalimentación desde el campo de aplicación puede conducir al descubrimiento de nuevas amenazas sobre la supervivencia del sistema o puede revelar deficiencias.
Resultados de periódicos testeos de penetración y evaluaciones de supervivencia (SSAs)	Resultados problemáticos obtenidos por el testeo de penetración del "*red team*" planificados periódicamente y las evaluaciones de seguridad/supervivencia pueden que se reconozca la necesidad de mejoras evolutivas.

Tabla 6. Elementos disparadores de actividades de diseño evolutivo para los sistemas con capacidad de supervivencia.

Actividad de Diseño Evolutivo	Ejemplo
No se necesita tomar ninguna acción	La gestión ejecutiva no ve ninguna nueva amenaza sobre la misión del sistema impuesta por el fuerte aumento en la contratación de empleados, debido a que los nuevos contratados son pasibles de minuciosos chequeos de conocimiento.
No se toma ninguna acción, pero se incrementa el monitoreo de este disparador (o conjunto de disparadores)	Se incrementa la cantidad de recursos destinados al monitoreo del retorno desde el campo, en respuesta a evidencias del sector operacional que indican una reducción del desempeño debido a una extraña combinación de acciones de los clientes.
Se necesita un ulterior análisis, generar escenarios par aun SSA o llevar adelante un testeo de penetración para determinar la próxima actividad, si la hubiera	Crear nuevos escenarios que reflejen los patrones de utilización de un nuevo tipo de cliente. Usar estos escenarios para realizar un SSA, cuyos resultados pueden dirigir nuevas actividades de diseño evolutivo.
Ejecutar una parte (delta) de una o más actividades del ciclo de vida de supervivencia	Un pequeño cambio en la arquitectura del sistema le incrementa la resistencia a nuevo escenario de ataque.
Realizar una porción (delta) de cada uno de los conjuntos completos de las actividades del ciclo de vida de supervivencia	Una modificación en la misión afecta en algún grado a todas las actividades del ciclo de vida.
Abandonar del sistema y realizar un rediseño completo	Un cambio principal en la tecnología en el dominio de aplicación, acompañado de mejoras totales en la tecnología defensiva, no puede ser incorporado sólo por las actividades de diseño evolutivo.
Abandonar el sistema	Un cambio drástico en la misión torna obsoleto al sistema.

Tabla 7. Posibles actividades de diseño evolutivo en respuesta a un evento disparador.

3.5 ACTIVIDADES DEL CICLO DE VIDA DE DISEÑO DE SISTEMAS BASADOS EN COTS -CBS-

Históricamente, la seguridad computacional ha estado basada en el uso de un conjunto de herramientas genéricas y soluciones que proporcionan una fortificación o defensa perimetral para las aplicaciones que están siendo protegidas. En la mayoría de los casos, estas soluciones de seguridad fueron agregadas o ideas tardías. Más aún, el ambiente abierto, ilimitado, altamente vulnerable y altamente colaborativo que presenta la Internet vuelve altamente inefectivos estos modelos de fortaleza. ***La supervivencia puede ser pensada como una solución de ingeniería de software que integra la seguridad computacional dentro del proceso de diseño y desarrollo de software desde su inicio. Protege la misión específica de la misión, proporciona el reconocimiento de los problemas que no pueden ser evitados totalmente, y provee esquemas de recuperación cuando los ataques (o los accidentes o las fallas de componentes) no pueden ser completamente evitados. El equilibrio entre gestión de riesgo e ingeniería son parte inherente del proceso de desarrollo para los sistemas con capacidad de supervivencia.***

Por lo tanto, las estrategias de supervivencia deben estar integradas a todo el ciclo de vida del desarrollo del software. Esto impone un reto particularmente fuerte al desarrollo de software basado en COTS, ya que las implicancias del desarrollo de este tipo de sistemas basado en COTS (*COTS-Based System -CBS-*) son desalentadoras. Con el modelo de fortaleza o basado en el perímetro, un CBS se debería desarrollar con poca o ninguna atención de la seguridad computacional, y al que luego se le debería agregar una defensa perimetral basada en COTS (por ejemplo, un *firewall* comercial más un paquete de encriptación, etc.) para mejorar la seguridad. Sin embargo, la capacidad de supervivencia es una propiedad global que emerge de las interacciones entre los componentes del sistema y resulta dificultoso discernir cuándo las cualidades intrínsecas son completamente conocidas. *Con los COTS, muchos de los atributos de calidad de software son desconocidos y difíciles de analizar sin tener acceso al código fuente u otros artefactos del proceso de ingeniería de software*.

Metas a largo plazo exigen de la creación de metodologías prácticas de desarrollo de software para la construcción de CBSs seguros y con capacidad de supervivencia. Se prevé trabajar en dos áreas complementarias de investigación de ingeniería de software: (1) la supervivencia para las actividades de desarrollo de software tradicional (es decir, a medida), y (2) actividades del ciclo de vida del CBS.

Al principio del presente trabajo se analizó la investigación sobre las metodologías de desarrollo de sistemas con supervivencia focalizado en el ciclo de vida de desarrollo de software tradicio-

nal, y asociado con las actividades de supervivencia. Desafortunadamente, estas *actividades del ciclo de vida tradicional (enriquecido con los elementos de supervivencia) no puede ser aplicado directamente al desarrollo de sistemas basados en COTS. Como ya se planteara, los CBSs imponen desafíos especiales y severos a cualquier equipo de desarrollo de software.* El mayor desafío es el de lidiar con información extremadamente limitadas acerca de los atributos de calidad del software de los productos COTS que se encuentran bajo consideración para ser utilizados como componentes del sistema. Generalmente, ninguno de los artefactos del proceso de ingeniería de software tradicional (código fuente, lógica de diseño, ambientes de testeo y resultados de testeo, etc.) son fáciles de obtener; para el caso de un típico producto COTS resulta casi imposible discernir qué criterios se aplicaron (explícita o implícitamente) a nivel de ingeniería del software sobre los diferentes componentes de la calidad (desempeño, seguridad, disponibilidad, mantenibilidad, usabilidad, etc.). Resulta innecesario decir que ésta es una desventaja severa para un equipo de desarrollo que está tratando de construir un sistema con capacidad de supervivencia a partir de componentes COTS.

Un paso importante es una extensión del modelo de ciclo de vida en espiral para incorporar las actividades de supervivencia. Esto se muestra en la Figura 14.

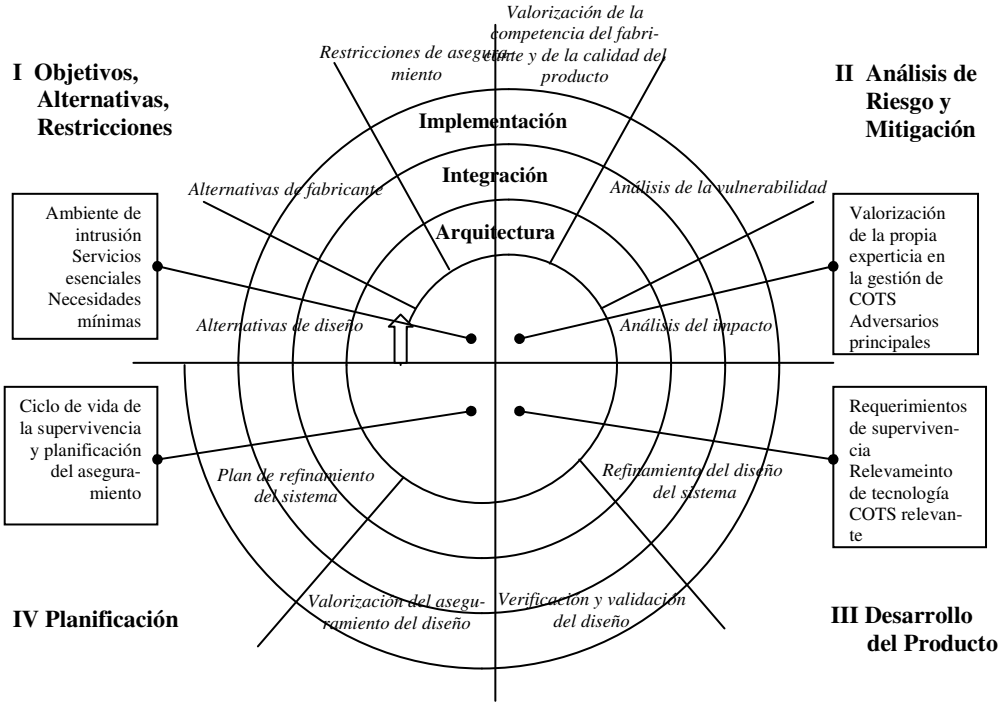


Figura 14. Modelo de ciclo de vida en espiral con actividades de supervivencia.

Imponerle al desarrollo de CBSs un proceso de ingeniería de software basado en principios ha sido el centro de la investigación inicial en las actividades del ciclo de vida de sistemas basados

en COTS [Brownsword 00, Oberndorf 00]. Si bien los aspectos relativos con la seguridad y la supervivencia no fueron tratados explícitamente en los primeros trabajos referidos, dicha investigación sirve como un basamento (junto con la investigación actual sobre la supervivencia para las actividades del ciclo de vida tradicional) en el desarrollo de una metodología para la construcción de CBSs con capacidad supervivencia.

El ciclo de vida de los CBSs incluye cuatro áreas de actividad, cada una de las cuales tiene varias áreas [Oberndorf 00]:

1. **Área de Actividad de Ingeniería**
   Contexto del Sistema, Arquitectura y Diseño, Mercado, Construcción, Gestión de la Configuración, Implantación y Soporte, Evaluación.
2. **Área de Actividad del Negocio**
   Caso del Negocio del COTS, Estimación del Costo del COTS, Relaciones con el Proveedor Interno, Relaciones con el Fabricante.
3. **Área de Actividad de Alcance del Proyecto**
   Estrategia CBS, Gestión del Riesgo del COTS, Balanceo CBS, Compartición de Información, Transición Cultural.
4. **Área de Actividad del Contrato**
   Requerimientos del Contrato, Seguimiento y Supervisión del Contrato, Solicitud, Negociación del Licenciamiento.

A su vez, *cada sub-área incluye un conjunto de actividades*. El conjunto completo de actividades se muestra en la Tabla 8. (Observar que las actividades dentro del área de actividad no son secuenciales excepto cuando se lo indique).

3.6 ACTIVIDADES DEL CICLO DE VIDA DEL DESARROLLO UTILIZANDO COTS Y SUPERVIVENCIA

Para los CBSs, una estrategia de supervivencia puede proporcionar el *framework* para que un conjunto específico de actividades de supervivencia sea asociado a las actividades de ciclo de vida descripto anteriormente en la Tabla 8. La estrategia debería proporcionar un *framework* para las actividades que se espera tengan lugar tanto en el proceso como en el producto. Por ejemplo, esta estrategia debería contener tanto la política como los problemas técnicos. Se deberían realizar estudios de factibilidad para determinar si los productos COTS satisfacen los requerimientos de supervivencia, como así también examinar los planes de evolución del fabricante para determinar si los productos COTS que actualmente satisfacen los requerimientos de supervivencia evolucionarán de tal manera que continúen haciéndolo.

Área de Actividad de Ingeniería	Actividades
Contexto del Sistema	Determinar y priorizar los elementos negociables y no-negociables del contexto del sistema. Comprender los elementos esenciales de los procesos de negocio de los usuarios finales. Modificar los procesos del usuario final. Negociar cambios en el contexto del sistema. Reflejar los resultados de los balanceos. Re-examinar periódicamente los procesos del negocio.
Arquitectura y Diseño	Seleccionar productos candidatos. Crear y evolucionar la representación de la arquitectura y el diseño. Validar la arquitectura. Reflejar los resultados de los balanceos. Comprender y reflejar el impacto del mercado.
Mercado	Crear y mantener actualizado el conocimiento del mercado disponible y emergente. Re-explotar el mercado. Alertar al equipo técnico sobre nuevas tecnologías prometedoras.
Construcción (incluye codificación, integración, testeo)	Descubrir y describir las características del producto. Crear el *glue code*[11]. Integrar y testear el sistema de manera temprana y continua. Determinar continuamente el efecto de las actualizaciones de producto.
Gestión de la Configuración	Identificar las líneas base de la configuración. Recibir y procesar las actualizaciones. Controlar de manera sistemática los cambios. Liberar nuevas versiones del sistema. Coordinar con construcción.

[11] También conocido como *glueware* o *binding code*. Es el nuevo código que se necesita para que el componente COTS pueda ser integrado a un sistema más grande. Generalmente se lo define como conectado al propio COTS, actuando como un "puente" ente el componente COTS y el sistema dentro del cual el mismo ha de ser integrado. Se lo puede requerir para que conecta un componente COTS ya sea con un código del sistema de alto nivel, o con otros componentes COTS utilizados dentro del sistema. Puede ser considerado de alguna de las siguientes formas: (1) cualquier código requerido para facilitar el intercambio de información o de datos entre el componente COTS y la aplicación; (2) cualquier código que se necesite para "enganchar" el componente COTS dentro de la aplicación, aún cuando no se lo necesite para facilitar el intercambio de datos; (3) cualquier código que se requiera para proveer la funcionalidad que originalmente esperada del componente COTS, y la que debe interactuar con ese componente COTS.

Implantación y Soporte	Planificar el soporte. Planificar las implantaciones del sistema. Planificar y adecuar las necesidades de soporte del usuario final. Incorporar los nuevos *releases* del producto. Coordinar con proveedores. Gestionar licencias. Realizar los ajustes específicos del sitio. Planificar y gestionar múltiples *releases*. Coordinar y ejecutar la ingeniería de los *releases* de múltiples proveedores con los *releases* propios.
Evaluación	Evaluar del plan. Evaluar del diseño. Localizar potenciales candidatos relevantes. Realizar los análisis apropiados para la selección de tecnologías o productos. Documentar y compartir información adquirida.
Área de Actividad del Negocio	**Actividades**
Caso de Negocio del COTS (estas actividades son secuenciales)	Determinar factores de éxito críticos para el sistema. Conducir un estudio de factibilidad preliminar. Identificar suposiciones claves del CBS Articular las alternativas. Formular planes estratégicos del CBS. Analizar alternativas. Volver a tratar el caso de negocio del COTS.
Estimación de Costo del COTS	Identificar los factores de costo. Seleccionar y calibrar los modelos de estimación del costo del COTS. Estimar costos. Proveer estimaciones de costo para sostener los otros conjuntos de actividades. Hacer el seguimiento de los costos reales vs. los estimados. Mantener los modelos de estimación de costos del COTS.
Relaciones con los Proveedores Internos	(Destinado a organismos gubernamentales)

Relaciones con el Fabricante	Comprender y monitorear la solución de largo plazo del fabricante relativa al mantenimiento y soporte. Desarrollar una estrategia para crear y gestionar las relaciones con el fabricante. Organizar reuniones e intercambios con los fabricantes. Establecer relaciones con otros clientes del fabricante. Alentar y facilitar relaciones de trabajo entre fabricante.
Área de Actividad de Alcance de Proyecto	**Actividades**
Estrategia del CBS	Identificar los objetivos, restricciones y presupuestos del CBS. Identificar los riesgos relacionados con el COTS. Identificar los segmentos de mercado relevantes. Identificar las soluciones alternativas basadas en COTS. Valorizar, evaluar y comparar las soluciones basadas en COTS. Recomendar una estrategia global del CBS. Crear un plan correspondiente al CBS. Revalorizar y revisar la estrategia y el plan de adquisición.
Gestión del Riesgo del COTS	Identificar y priorizar los riesgos relacionados con el COTS. Analizar los riesgos relacionados con el COTS. Planificar e instituir mitigaciones del riesgo del COTS. Hacer el seguimiento de los riesgos relacionados con el COTS. Revisar regularmente el éxito de la gestión de riesgos del COTS.
Balanceo del CBS	Determinar la organización y los roles del contratista. Identificar dónde resulta necesario un balanceo de CBS. Reunir suficiente información para un balanceo relacionado con COTS. Seleccionar o tomar una resolución apropiada del CBS. Comunicar la resolución.
Compartir Información	Determinar el conjunto de información y estrategias compartidas. Monitorear activamente el uso de la información provista para compartir. Buscar información de CBS en fuentes externas. Asegurar la recolección de información de CBS. Dejar accesible a los demás nuestra información de CBS. Incluir información compartida en nuestros procesos.

Transición Cultural	Valorizar la disponibilidad del CBS en la organización. Identificar el conjunto de destrezas requeridas para el éxito del CBS. Entrenar a quien lo necesite. Asegurar el compromiso con el CBS de ejecutivos seniors. Identificar y alentar defensores de CBS. Proporcionar incentivos para el cambio. Compartir información.
Área de Actividades del Contrato	**Actividades**
Requerimientos del Contrato (destrezas necesarias)	Establecer requerimientos específicos del COTS dentro de los requerimientos del contrato. Estimar las solicitudes de cambios de contrato para determinar sus efectos sobre los productos COTS.
Seguimiento y Supervisión del Contrato	Utilizar bancos de prueba y ensayos piloto que provean visibilidad. Involucrar a la comunidad de usuarios finales en las pruebas piloto.
Solicitud	Preparar estimaciones de costo y de planificación para los productos. Prepararse para la evaluación de las respuestas. Conducir la evaluación de la propuesta.
Negociación del Licenciamiento	Realizar una investigación preliminar de las alternativas de licenciamiento y sus costos. Asegurar un presupuesto. Negociar las licencias.

Tabla 8. Actividades del Ciclo de Vida.

Área de Actividad de Ingeniería	Actividades de supervivencia
Contexto del Sistema	Comprender nuestra misión del negocio y sus consecuencias en términos de la supervivencia, los requerimientos de supervivencia y los servicios esenciales. Comprender las restricciones tales como las redes existentes, los problemas de gestión, etc. Comprender las estrategias de supervivencia de los demás sistemas externos en este momento. Periódicamente re-examinar el contexto y los requerimientos de supervivencia, y los procesos de negocio asociados con la supervivencia, y trazar los cambios.
	Política: Desarrollar o modificar la política global para incluir los aspectos de supervivencia.
Arquitectura y Diseño de la Organización	Refinar la estrategia global de supervivencia dentro del área de Arquitectura. Emplear *Survivable Systems Analysis*, *Survivable Network Design*. Comprender las consecuencias respecto de la supervivencia de los productos seleccionados. Incorporar las capacidades de supervivencia de los productos seleccionados. Considerar las vulnerabilidades por fuera de los componentes específicos que pueden ser parte del proceso normal, tanto en los sistemas como en las operaciones.
	Política: Considerar los procesos de negocio que soporta la tecnología y que son esenciales para la supervivencia.
Mercado	Mantenerse al día de las nuevas técnicas de supervivencia. Revisar el mercado con la supervivencia en mente. Alertar al equipo sobre las consecuencias y las capacidades de supervivencia de nuevas tecnologías.
Construcción (incluye codificación, integración, testeo)	Utilizar estrategias de codificación defensiva, verificación de corrección, testeo de penetración, y testeo estadístico. Determinar de manera continua el impacto de las actualizaciones de productos relacionadas con la supervivencia. Considerar la integración y la interoperabilidad relativa con la supervivencia. Considerar la ejecución a medida y su impacto sobre la supervivencia. Tener en cuenta la preservación de las propiedades. Desarrollar una discusión acerca de la supervivencia.
	Política

Gestión de la Configuración	Asegurar que los cambios y las actualizaciones no impacten negativamente sobre la supervivencia. Utilizar un esquema de gestión de configuración que haga visibles aspectos de la supervivencia.
	Política
Implantación y Sostenimiento	Establecer una actividad de observación de la supervivencia y un equipo de evaluación del riesgo de la supervivencia. Considerar la evolución del producto del fabricante, la evolución de la tecnología y la evolución del sistema. Examinar los nuevos productos y los nuevos *releases* para la supervivencia. Analizar la evolución a largo plazo y sus consecuencias sobre la supervivencia; mantener y mejorar la supervivencia. Ajustar/reaccionar a las decisiones técnicas que han hecho los *partners* y clientes.
Evaluación	Valorizar el éxito de la estrategia de supervivencia. Realizar análisis de supervivencia para la selección de tecnologías y productos. Documentar y compartir la información adquirida.
Área de Actividad del Negocio	**Actividades de supervivencia**
Caso de Negocio del COTS	Valorizar qué COTS extendido puede suportar las características de supervivencia requeridas. Valorizar si uno puede neutralizar efectos colaterales indeseados, por ejemplo, actualizaciones automáticas realizadas por el fabricante. Valorizar la duplicación del esfuerzo/interoperabilidad desde el punto de vista de la supervivencia (por ejemplo, ¿el fabricante requiere de archivos de contraseñas separadas que necesita de un mantenimiento separado?) Valorizar el costo del impacto de ataques. Determinar el éxito de los factores críticos para el sistema. Comprender las implicancias financieras y revisar el caso de negocio. Revisar el caso de negocio si cambian los factores críticos de análisis de sensibilidad. Esto se aplica tanto al sistema como a los procesos.
	Política
Estimación de Costo del COTS	Utilizar la supervivencia como un factor del costo en los modelos de estimación de costos seleccionados. Estimar el impacto del costo de la construcción de la supervivencia o de la adquisición de productos COTS para la supervivencia en el ambiente de amenaza.

Relaciones con los Proveedores Internos	
Relaciones con el Fabricante	Desarrollar una estrategia para valorizar fabricantes en relación con la supervivencia. Alentar y facilitar discusiones relacionadas con la supervivencia entre fabricantes. ¿Qué tan bien posicionados están los fabricantes en relación a la dirección donde deseamos ir? ¿Continuarán teniendo presencia en este campo en el largo plazo?
Área de Actividad de Alcance de Proyecto	**Actividades de supervivencia**
Estrategia del CBS	Desarrollar una estrategia de supervivencia global. Desarrollar un plan de supervivencia. Identificar las características de supervivencia necesarias. Examinar soluciones basadas en COTS para la supervivencia alternativas.
Gestión del Riesgo del COTS	Aplicar OCTAVE como parte del esquema de gestión del riesgo. Hacer el seguimiento de los riesgos de supervivencia además de los riesgos globales del sistema.
Balanceo del CBS	Valorizar las características de supervivencia de los productos COTS considerados. Balancear la supervivencia contra los atributos requeridos en el contexto del proyecto.
Compartir Información	Reunir información relativa al ambiente de amenaza y a la supervivencia de productos CBS y hacerla accesible a los demás. Incluir participantes y equipo técnico.
Transición Cultural	Identificar a los destacados en temas de supervivencia y asegurar la contratación de participantes a las necesidades de la supervivencia. Proporcionar entrenamiento informativo sobre supervivencia a todo el personal y en profundidad a los que lo necesiten. Recordar que la transición de la seguridad a la supervivencia puede resultar amenazante para el equipo de seguridad tradicional, y obtener su participación.
Área de Actividades del Contrato	**Actividades de supervivencia**
Requerimientos del Contrato (destrezas necesarias)	Considerar la experiencia en el campo de la supervivencia del fabricante y de otros contratistas (por ejemplo, integradores, en red), y especificar la supervivencia en los requerimientos del contrato.

Seguimiento y Supervisión del Contrato	Obtener visibilidad en la supervivencia de los productos COTS. Tener la capacidad de ajustar los contratos para reflejar los cambios de la supervivencia, en particular en lo relativo a las amenazas. Monitorear el desempeño del fabricante de COTS en relación con los requerimientos de supervivencia. Obligar a los contratistas a compartir los riesgos. Tener la habilidad de auditar/valorizar los sistemas del contratista.
Solicitud	Establecer criterios de evaluación de la supervivencia de los productos y servicios provistos por el fabricante. Incluir la supervivencia en las estimaciones de costo y planificación y en los criterios de evaluación.
Negociación del Licenciamiento	Determinar si la supervivencia debería ser parte del acuerdo de licenciamiento. Evaluar las expectativas de supervivencia en caso de expiración o cambio.

Tabla 9. Actividades del Ciclo de Vida de COTS a al medida de la supervivencia.

3.6.1 Actividades de supervivencia en CBS

Se han introducido algunas modificaciones en las áreas de actividad del CBS e incluido mucha información en una matriz con las actividades de supervivencia asociadas [Mead 01]. Las áreas y sub-áreas de actividades CBS ligeramente revisadas son:

1. **Área de Actividad de Ingeniería**
   Contexto del Sistema, Arquitectura y Diseño de la Organización, Mercado, Construcción, Gestión de la Configuración, Implantación y Sostenimiento, Evaluación.
2. **Área de Actividad de Negocio**
   Caso de Negocio del COTS, Estimación de Costo del COTS, Relaciones con el Fabricante
3. **Área de Actividad de Alcance del Proyecto**
   Estrategia CBS, Gestión del Riesgo del COTS, Balanceo CBS, Compartición de Información, Transición Cultural, Política.
4. **Área de Actividad del Contrato**
   Requerimientos del Contrato (destrezas requeridas), Seguimiento y Supervisión del Contrato, Solicitud, Negociación del Licenciamiento

Un conjunto completo de las áreas de actividades CBS suplementadas con las actividades de supervivencia se muestra en la Tabla 9. En el Anexo 2 se presentan, a modo de ejemplo, algunos detalles en la sub-área Contexto del sistema.

3.7 OPORTUNIDADES DE FUTURAS INVESTIGACIONES

Son posibles muchas extensiones de este trabajo. Podría ser investigado el contexto más amplio de la supervivencia, el sistema, y los modelos de ciclo de vida basados en COTS y sus actividades asociadas. Por ejemplo, se podrían expandir y refinar las actividades de supervivencia del CBS, y reflejar luego los cambios en las áreas y sub-áreas de actividades CBS. Cada una de estas actividades podrían se descriptas con mucho mayor detalle, acompañadas de ejemplos y casos de estudio, para proporcionar un *framework* práctico para la construcción de CBS con supervivencia. Esto le podría permitir a los desarrolladores de CBS comenzar a mejorar y sostener la supervivencia del sistema. Esta investigación de la supervivencia de CBS es parte de una actividad de investigación global en el campo de los métodos relativos con la supervivencia del sistema que puedan ser incorporados en las diferentes fases del ciclo de vida.

Algunos de los métodos referenciados en las áreas de actividad son parte del futuro plan de investigación. Un siguiente paso clave para la evolución de la supervivencia es el desarrollo de abstracciones y métodos de razonamiento más poderosos para la definición del comportamiento

y estructura de sistemas distribuidos de gran escala. Tales resultados posibilitarán un análisis más comprensivo de los servicios esenciales y de las trazas de intrusión al limitar la complejidad. Además, se requiere del mejoramiento de las representaciones y los métodos para la definición de intrusiones. Resulta importante avanzar más allá de las limitaciones del lenguaje natural y desarrollar semánticas uniformes para la utilización de intrusión, que permita un análisis más riguroso e incluso permitir su aplicación en métodos computacionales.

Otra línea de investigación fructífera comprende el desarrollo de estilos o plantillas arquitectónicas estandarizadas para las estrategias de supervivencia que puedan ser insertadas y combinadas con las arquitecturas del sistema para mejorar sus propiedades de supervivencia. Tales plantillas pueden ser analizadas en forma independiente para definir y documentar su contribución a la supervivencia del sistema.

SECCION 4: EL MODELO TRUSTWORTHY REFINEMENT THROUGH INTRUSION-AWARE DESIGN - TRIAD

Robert J. Ellison y Andrew P. Moore publican hacia finales del año 2002 el trabajo "*Trustworthy Refinement Through Intrusion-Aware Design (TRIAD)*", y su posterior revisión hacia principios del año 2003. Este trabajo se presenta como un corolario de las investigaciones iniciadas en el año 1997 a través del reporte "*Survivable Network Systems: An Emerging Discipline*", seguidos por los reportes "*Survivable Network Analysis Method*", publicado en el año 2000, y "*Modelos de Ciclo de Vida para Sistemas con Capacidad de Supervivencia*", publicado en el año 2002. Este conjunto van planteando respuestas progresivas en su alcance a este nuevo campo de la ingeniería de desarrollo de software.

De esta manera se afronta *la necesidad de replantear un conjunto de conceptos y de metodologías relacionados con los ciclos de vida de los sistemas de información de cara a las nuevas realidades tecnológicas y socios-culturales, que constituyen verdaderos retos para los profesionales e investigadores*.

Una alta confianza en la supervivencia de un sistema requiere de un conocimiento preciso del ambiente de amenazas del sistema y del impacto de dicho ambiente sobre las operaciones del sistema. Desafortunadamente, los actuales métodos de desarrollo de sistemas de información seguros y con supervivencia a menudo emplean una solución relativamente inconsistente en la cual se coloca el centro de atención en la decidir cuáles componentes de seguridad de uso frecuente resulta necesario integrar en lugar de realizar una valoración racional de cómo identificar los ataques que probablemente hayan de comprometer la misión global.

Por esta razón, se propone un modelo de diseño de alerta de intrusión denominado *Trusworthy Refinemet trough Intrusion-Aware Design – TRIAD*.

TRIAD ayuda a quienes toman decisiones en los sistemas de información a formular y mantener una estrategia de supervivencia coherente, justificable y accesible para identificar las amenazas que comprometen la misión de su organización. TRIAD también ayuda a la evaluación y mantenimiento del diseño de un sistema de información en términos de su habilidad de implementar una estrategia de supervivencia. (Se ha incluido un caso de aplicación en el refinamiento de una estrategia de supervivencia para un negocio que vende productos a través de Internet).

TRIAD proporciona una sólida base para el posterior refinamiento, experimentación y validación de una solución que explote el conocimiento que se posea acerca del comportamiento del intruso para mejorar el diseño de la arquitectura y las operaciones del sistema. Finalmente, con una efectiva herramienta de soporte y la evidencia de su eficacia TRIAD será integra-

do como modelo del ciclo de vida más integral aplicable al desarrollo y mantenimiento de sistemas de alta confianza.

4.1 INTRODUCCION

Los proyectistas e ingenieros de sistemas de información crítica enfrentan con grandes dudas los retos de adquirir, desarrollar y mantener sistemas complejos interconectados que aseguren el éxito de la misión a pesar del creciente número de ataques, cada vez más sofisticados, a las redes de computadoras. Quienes toman decisiones deben elegir dentro una importante variedad de tecnologías de información al momento de considerar la multitud de vulnerabilidades crecientemente explotadas por una comunidad de atacantes maliciosa y coordinada. Desafortunadamente, los modelos existentes para la construcción de sistemas de información seguros resultan limitados en cuanto a guiar a los encargados de la toma de decisiones hacia soluciones coherentes e integrales que sean tanto efectivas como accesibles.

Los métodos de desarrollo de sistemas seguros generalmente estimulan soluciones aisladas para identificar problemas particulares, dando por resultado diseños inconsistentes que raramente resultan robustos para enfrentar un ataque malicioso. Tales soluciones resultan desacopladas respecto de los riesgos que intentan identificar, oscureciendo la justificación de su aplicación. Generalmente resulta o en un sobre-dimensionamiento –en donde las soluciones sugeridas son más fuertes, menos eficientes y más costosas de lo necesario- o en un sub-dimensionamiento –en donde las soluciones no identifican adecuadamente las amenazas a la misión relevante. Parte del problema es la existencia de técnicas que se centran casi exclusivamente en el diseño *bottom-up* de los sistemas, partiendo de los componentes existentes y, así, perdiendo de vista la misión global. Existe un escaso conocimiento real de cómo los ataques han de afectar la supervivencia de aquello que es importante para la organización en caso de que los mismos ocurran. *Los desarrolladores necesitan definir una estrategia de supervivencia específica para los sistemas que construyen que describa la solución global para resistir, reconocer, recuperarse y adaptarse a los ataques que comprometen a la misión crítica.* Necesitan tanto de una solución táctica como estratégica, pero los mecanismos tácticos deben satisfacer a los objetivos estratégicos para asegurar el éxito de la misión.

Los administradores de los sistemas de información a menudo, no intencionalmente, desarrollan los sistemas bajo su administración en direcciones que resuelven problemas limitados y a corto plazo, de espaldas a los objetivos estratégicos y a la supervivencia de la misión. Una solución que puede resultar óptima localmente a menudo conduce por un camino que nos aleja de una solución globalmente óptima. Esto resulta análogo a ganar una batalla, pero perder la guerra.

Una mirada estrecha de los administradores sobre la resolución del "problema del día" no resulta sorprendente, dado que, aún en caso que exista una estrategia de supervivencia documentada, a menudo resulta difícil saber cuán bien un sistema de información implementa esa estrategia. Y exacerbando esta situación, el ambiente de amenazadas para los sistemas basados en Internet son extremadamente dinámicos, requiriendo de una re-evaluación regular de la supervivencia atenta a un cambio en la actividad del atacante o a una mejor comprensión de las amenazas percibidas.

La supervivencia de un sistema de alta confianza requiere de un conocimiento acertado del ambiente de amenazas del sistema y del impacto de dicho ambiente sobre las operaciones del mismo. Las técnicas reduccionistas que se especializan en el diseño e implementación de bajo nivel, que pierden de vista el ambiente global, están condenadas al fracaso. En respuesta a esta situación, se propone un modelo de diseño basado en la idea de *intrusion-aware* (*Intrusion-Aware Design - AID*) denominado *Trustworthy Refinement through Intrusion-Aware Design - TRIAD*.

TRIAD ayuda a quienes toman decisiones en un sistema de información a:

- ***Formular y mantener una estrategia de supervivencia coherente, justificable y accesible que identifique las amenazas que comprometen la misión de su organización***. La formulación de una estrategia de supervivencia ayuda a asegurar una base sólida para la adquisición y desarrollo, resultando necesario el mantenimiento de dicha estrategia a fin de asegurar que la misma permanece robusta frente a los cambios inevitables en el ambiente de amenazas.
- ***Evaluar y mantener un diseño de sistema de información en términos de su aptitud para implementar una estrategia de supervivencia***. La evaluación del diseño del sistema ayuda a asegurar que la estrategia está siendo adecuadamente implementada (tal aptitud será inestimable al momento de evaluar las ofertas frente la adquisición de sistemas), resultando necesario el mantenimiento del diseño para asegurar que continúa implementando la estrategia de cara a la inevitable evolución del diseño e implementación del sistema.

TRIAD ayuda a los ingenieros a comprender las complejas interacciones entre el sistema de información, su misión, y su ambiente de amenazas en todos los niveles del refinamiento en la arquitectura del sistema. Los sistemas de información incluyen la combinación de tecnología de información y las actividades de las personas que utilizan esa tecnología para atender a operaciones, administración y toma de decisiones. El reporte se centra particularmente en los sistemas de información interconectados, altamente distribuidos de gran escala, tales como aplicaciones basadas en Internet[12]. Los modernos sistemas de información basados en la combinación de

[12] De aquí en más, a menos que se indique lo contrario, nuestro uso del término "sistema" se refiere específicamente a este tipo de sistemas de información interconectado, altamente distribuido, de gran escala, los cuales incluyen la combinación de tanto la tecnología de información como su de contexto operacional.

computadora/red generalmente atraviesan los límites de la organización y carecen de una administración centralizada y de una política de seguridad unificada. Resulta imposible controlar, e incluso conocer, el número y naturaleza de los nodos conectados a los sistemas de información basados en una Internet sin límites definidos. La distinción entre persona de confianza e intruso puede variar en que un socio de una actividad puede ser un competidor o adversario para otro. La sociedad se está volviendo cada vez más vulnerable a las amenazas de alto impacto sobre los sistemas complejos e ilimitados. TRIAD le permite a los ingenieros de los sistemas de información utilizar patrones de ataque conocidos o hipotéticos y así poder mejorar iterativamente y mantener continuamente la supervivencia del sistemas, aún a medida que el ambiente de amenazas evolucione a lo largo del tiempo. TRIAD focaliza su atención en los patrones de ataque y en las estrategias de supervivencia al ataque a nivel de la arquitectura, para evitar ser agobiado por los detalles de las vulnerabilidades de cada componente particular o de soluciones de seguridad por fragmentos. En particular, se consideran ataques maliciosos, más que fallas o accidentes no-maliciosos, debido a la creciente sofisticación, frecuencia y severidad de tales ataques y a lo inadecuado de las soluciones existentes para enfrentarlos. Se centra la atención en aquellos ataques presumibles para el sistema de interés, en lugar de hacerlo sobre todos los ataques que resultan teóricamente posibles, a fin de asegurar la relación costo-eficiencia y la relevancia de la aplicación de TRIAD y de las soluciones que el mismo propone. Allí donde resulta posible, el modelo promueve el empleo de bloques constructivos asociados a la seguridad y a la supervivencia que ayuden a resistir, reconocer y responder dinámicamente ante los probables intrusiones. Se consideran tanto bloques constructivos tecnológicos como procedurales, dado que las soluciones tecnológicas particulares destinadas a problemas de supervivencia pueden no encontrarse disponibles, ser demasiado inmaduras, o demasiado costosas para la organización que está construyendo el sistema.

TRIAD facilita la planificación asociada con el inevitable cambio del ambiente de amenaza y el ambiente operativo y ayuda a la rastreabilidad de los efectos debido a estos cambios en lo que hace a los requerimientos de supervivencia y de arquitectura. En particular, se requiere de la rastreabilidad de las soluciones arquitectónicas referidas a las intrusiones que, se supone, dichas arquitecturas solucionan. La documentación de rastreabilidad resulta esencial con respecto a las modificaciones en el sistema causadas por los cambios en el perfil de riesgo de la organización, la aparición de nuevos patrones de ataque, la accesibilidad a nueva tecnologías capaces de soportar requerimientos tanto de funcionalidad como de seguridad, y los cambios en los procesos de trabajo subyacentes que afectan la vulnerabilidad y el análisis de riesgo.

Se describen los elementos principales, las relaciones fundamentales y las técnicas que dan soporte a TRIAD. ***El modelo no representa el proceso de desarrollo completo, sino aquella parte que***

tiene que ver con el refinamiento de la arquitectura sólo desde la perspectiva de la supervivencia. En particular, no se representan aquellas partes del proceso que requeridas para refinar la función del sistema en su forma más general o para considerar otros atributos de calidad distintos de la supervivencia. No obstante, ***este modelo provee una base sólida para el posterior refinamiento, experimentación y validación de una solución que explote el conocimiento del comportamiento del intruso para mejorar el diseño de la arquitectura del sistema y sus operaciones.***

4.1.1 Trasfondo

Los desarrolladores de muchas disciplinas de la ingeniería se basan en los datos de falla para mejorar sus diseños. Imaginemos lo que sucedería si los constructores de barcos hubieran ignorado las lecciones aprendidas acerca del inadecuado espacio de botes salvavidas que causara la gran cantidad de pérdidas de vidas cuando se hundiera el Titanic. El éxito en la ingeniería requiere que también nosotros aprendamos de desastres menos famosos.

Las empresas y los gobiernos históricamente han sido renuentes en revelar información acerca de fallas en la seguridad en sus sistemas debido a intrusiones, por temor a perder la confianza del público o a que otros atacantes pudiesen explotar la misma vulnerabilidad u otra similar. Sin embargo, el creciente interés del público y la cobertura periodística de los problemas de seguridad en la Internet ha conducido a un aumento en la publicación de datos de ataques en libros, grupos de noticias de Internet, y consejeros de seguridad tales como el CERT/CC. *Desafortunadamente, los desarrolladores de sistemas de información utilizan información sobre fallas de seguridad, es decir, intrusiones, sólo de una manera reactiva, para aplicar parches a los sistemas que ya se encuentran en producción, haciéndolo incluso de una manera muy incompleta e ineficiente* [Arbaugh 00]. Hoy en día, los sistemas de información que se construyen y se administran están propensos a las mismas o similares vulnerabilidades que los han plagado por años.

Retomando el concepto de supervivencia como la capacidad de un sistema de cumplir con su misión crítica vía la preservación de sus servicios esenciales, aún cuando dichos sistemas sean penetrados y comprometidos, resulta que dicho concepto resulta fuertemente dependiente de la flexibilidad de las estructuras del sistema de información, las que se deben planificar y construir durante los pasos iniciales del proceso de diseño del sistema. Bart Prakken describe acertadamente esta necesidad de flexibilidad:

> *La **estructura de las organizaciones** es comparable con el esqueleto de los vertebrados. Estas criaturas necesitan alguna solidez para sobrevivir dentro de un ambiente hostil. Sin embargo, si los vertebrados poseyeran demasiada estructura, por ejemplo debido a la artritis, su movilidad es entorpecida, con consecuencias que probablemente resulten fatales. Los mismos argumentos se aplican a la estructura de las organizaciones. Demasiada es-*

> *tructura disminuye la movilidad -flexibilidad- de una manera inaceptable. Y la flexibilidad resulta especialmente importante en ambientes hostiles. En consecuencia, las estructuras de las organizaciones (que contienen estructuras de información), con un bien considerado balance entre rigidez (estructura) y flexibilidad, son precondiciones necesarias para la creación de oportunidades de su supervivencia a largo plazo* [Prakken 00].

La flexibilidad resulta particularmente crucial para la supervivencia de los sistemas de información basados en Internet, a los fines de asegurar la disponibilidad de respuestas adecuadas a un ambiente de amenazas rápidamente cambiante.

La supervivencia requiere de una estrategia para reconocer, recuperarse y adaptarse a las intrusiones, y en la medida de lo posible, antes que nada, prevenir intrusiones. Las propiedades de supervivencia generalmente emergen de la interacción de los componentes del sistema, y, por lo tanto, deben ser consideradas al inicio de su proceso de desarrollo [Fisher 99]. *La consideración demasiado tardía de la supervivencia puede conducir a un diseño del sistema que embeba vulnerabilidades, haciendo luego muy difícil, muy costoso o directamente imposible incluir implementaciones de supervivencia en dicho diseño. El diseño de un sistema con supervivencia evoluciona, no de manera casual, sino a través de una planificación intuitiva que posea la flexibilidad necesaria para responder a las amenazas más probables. Los procesos de trabajo del negocio de una organización, incluidas la operación y la administración de la tecnología que dan sustento a dichos procesos, resultan absolutamente esenciales para la supervivencia de la misión de la organización.*

TRIAD ha sido construido a partir de la premisa de que resulta necesario el empleo de una manera mucho más pro-activa de la información disponible sobre ataques para la construcción de sistemas de costo efectiva que puedan sobrevivir a dichos ataques con un alto grado de certidumbre. La construcción de arquitecturas de supervivencia accesibles demanda el conocimiento del ambiente de amenazas del sistema de tal manera que ese esfuerzo se aplica sobre las intrusiones más probables más que en todas aquellas posibles.

Desafortunadamente, gran parte de la información sobre ataques que se encuentra disponible es muy detallada en términos de versiones de software, configuraciones específicas de una organización y scripts específicos del ataque. Tales detalles poseen una vida relativamente corta dado que los atacantes crean y modifican sus herramientas y métodos. Sin embargo, los patrones de ataque son mucho más constantes a lo largo del tiempo. Los ataques pueden tener por objetivo personas, procesos y estructuras físicas, como así también la tecnología del sistema. Del mismo modo, una estrategia de supervivencia puede contar con remedios procedurales, físicos y tecnológicos destinados a las vulnerabilidades de la misión que aseguren la viabilidad y

accesibilidad del remedio [Anderson 01].

Lograr tener confianza en la supervivencia de un sistema requiere dar evidencias de que el sistema es lo adecuadamente elástico para los patrones de ataque más probables. La naturaleza dinámica del ambiente de intrusión demanda de que TRIAD, y de las técnicas de análisis en la que está basada, ayudan a descubrir y a hacer hipótesis acerca de las nuevas fuentes y patrones de ataques, además de estar informados de los ataques provenientes de los adversarios conocidos. *Los patrones de ataques describen las estrategias generales del ataque, tal como las diversas formas de los ataques de denegación de servicio, y se pueden estructurar de tal manera que puedan aplicarse en una variedad de contextos* [Moore 01a]. La experiencia del Centro de Coordinación del CERT (CERT/CC) en el análisis de supervivencia de sistemas reales en ambientes industriales y gubernamentales y la recopilación de datos referidos a ataques basados en Internet conducen a un conocimiento más profundo de los patrones de ataques, tendencias y contramedidas [Ellison 99a, CERT 02].

4.1.2 Trabajos relacionados

Se están comenzando a realizar algunos pocos esfuerzos destinados a lograr el mejoramiento de los métodos de desarrollo de la seguridad y supervivencia de los sistemas interconectados centrados en el ambiente de amenazas. Neumann ofrece una importante comprensión y una visión de conjunto de los mecanismos que sustentan el desarrollo de arquitecturas de sistemas con supervivencia [Neumann 00]. El IATF (*Information Assurance Technical Framework - IATF*) incluye líneas directivas de gran alcance para la selección de los mecanismos de seguridad factibles de incorporar en sistemas de misión crítica de gran escala, basados en una caracterización de alto nivel de la amenaza y del valor de la información protegida [IATF 02]. Otro material en el que se bosqueja una metodología de ingeniería de un sistema seguro basada en un análisis más amplio del ambiente de amenazas, y está, en consecuencia, de alguna manera alineado con esta solución es [Salter 98]. Otro trabajo dentro del área de tolerancia a intrusión, el que se basa fundamentalmente en DARPA de EEUU (*Defense Advanced Research Projects Agency*) y en el proyecto MAFTIA de Europa (*Malicious- and Accidental-Fault Tolerance for Internet Applications*), está focalizado en la intrusión y aborda la supervivencia de sistemas distribuidos de gran escala, pero esta investigación usualmente ha ignorado los ataques y las contramedidas no técnicas [MAFTIA 02].

A lo largo de los años se han definido un gran número de listados de bloques constructivos asociados con la seguridad de un sistema. ***TRIAD deriva una estrategia de supervivencia haciendo uso de lo que denominamos tácticas de supervivencia. Las tácticas de supervivencia son un clase especial de táctica de arquitectura, que emplean métodos de diseño basados en atributos desarrollados en el Software Engineering Institute, los que se utilizan para dar sustento***

a atributos de calidad dentro de una arquitectura [Bachmann 02]. Otros trabajos recientes realizados por Anderson y Ramachandran merecen especial atención, el último de los cuales describe primitivas aprovechables dentro del contexto del diseño de arquitectura [Anderson 01, Ramachandran 02]. El IATF antes mencionado, también presenta un gran número de bloques constructivos asociados con la seguridad y la supervivencia que se clasifican de acuerdo al grado de protección ante ataques maliciosos. Dependiendo del nivel de amenaza que pueda esperarse y de las propiedades requeridas para el dominio de aplicación, la estructura recomienda mecanismos específicos y la confianza requerida a dichos mecanismos.

Existen numerosas fuentes en la extensa literatura sobre seguridad y supervivencia referida a la aptitud de las tácticas que aseguren el éxito de la misión. Por ejemplo, la *RAND Corporation* ha publicado un método para mejorar la supervivencia de los sistemas que se basa en categorías predefinidas de vulnerabilidades y técnicas de supervivencia [Anderson 99]. Si bien el método de *RAND* no se ha aplicado en gran medida, el estudio examina un amplio rango de sistemas existentes y de logros en el campo de la investigación sobre seguridad y supervivencia como para derivar categorías de vulnerabilidades y técnicas. Las técnicas de supervivencia identificadas proporcionan un buen punto de partida para identificar las técnicas para el diseño de sistemas con capacidad de supervivencia que resulta de utilidad para IAD. Otros trabajos sobre arquitecturas de supervivencia que también proporciona información de utilidad para el proceso IAD son los de Knight y Neumann [Knight 00b, Neumann 00].

Un amplio rango de trabajos sobre análisis de riesgo de la seguridad contribuyeron a este esfuerzo, incluidas las áreas de modelado de adversario, especificación de ataque, análisis de vulnerabilidad/amenaza, bases de datos y taxonomía relacionadas con la seguridad, análisis de impacto y *red-team*. El análisis de riesgo de la seguridad comprende el análisis de las amenazas al sistema y las vulnerabilidades y su potencia impacto en la misión del sistema. Los tres principales elementos de riesgo se pueden definir de la siguiente manera [DoD 99, DoD 00]:

- **Amenaza** Cualquier circunstancia o evento con la potencialidad de causar daño a un sistema.
- **Vulnerabilidad** Una característica del sistema que podría ser explotada por una amenaza para dañar a un sistema.
- **Impacto** La magnitud del daño causado a un sistema por una amenaza que explota una vulnerabilidad de un sistema.

El riesgo se define de una manera formal como "una combinación de la probabilidad de ocurrencia de una amenaza, la probabilidad de que la ocurrencia de una amenaza dé por resultado un impacto adverso, y la severidad del impacto resultante" [DITSCAP 99]. Entonces, en lo que

hace al contexto del trabajo, una amenaza maliciosa puede ser vista como cualquier actividad que explota una vulnerabilidad de un sistema y que provoca un impacto negativo en el éxito de la misión[13].

La experiencia a lo largo de los años en el análisis de riesgo de la seguridad recomienda un conjunto de peligros que evitar [Soo Hoo 00].

- **Complejidad** Muchas veces, las técnicas requieren la consideración explícita de todas las amenazas y vulnerabilidades, desde la más común hasta la más confusa, sin ningún tipo de filtrado que tenga en cuenta la probabilidad o el impacto. La complejidad resultante tiende a hacer agobiante el análisis.
- **Incompleto** A menudo, las técnicas ignoran aspectos claves del problema de administración del riesgo o hacen suposiciones incorrectas acerca del dominio del problema. Esto puede provocar, por ejemplo, que se enfaticen amenazas o soluciones tecnológicas por sobre aquéllas de procedimiento.
- **Indisponibilidad de datos** A veces, las técnicas requieren de la obtención de datos cuantitativos precisos sobre la probabilidad de las amenazas y de la severidad del impacto. En el mundo real, estos datos son recolectados y reportados de manera inconsistente, y con un alto grado de ambigüedad. El empleo de "estimaciones" altamente inciertas en un contexto en que se requieren datos precisos a menudo conduce a obvias fallas en los resultados, o, lo que es peor, a confusiones absurdas pero verosímiles.
- **Desajuste entre amenaza/contramedida** Las técnicas de administración del riesgo de la seguridad que se aíslan debido al empleo de tecnologías y prácticas de seguridad de uso popular, desconectadas de los objetivos de la misión o de las amenazas, tienden a desajustar las contramedidas del riesgo que las mismas supuestamente reducen. La falta de una rastreabilidad dificulta la evaluación segura del verdadero riesgo residual que resulta del uso de tecnologías y prácticas.
- **Análisis estático** Generalmente, las técnicas sólo tienen que ver con el ambiente de amenazas actual, y ponen escasa atención en la administración del sistema bajo amenazas cambiantes. Los rápidos cambios en el ambiente de amenaza, que son una característica de los modernos sistemas basados en Internet, demandan de técnicas que se puedan aplicar como parte de un ciclo de vida de diseño y mantenimiento evolutivo.

Claro está que no existen soluciones sencillas para estos problemas. Cada investigación en el análisis de riesgo de la seguridad generalmente promueve soluciones integrales que se tornan

[13] De aquí en adelante, "amenaza maliciosa" será referida simplemente como "amenaza", dado que éste es nuestro tema de interés principal. En el caso que sea necesario hacer la distinción, se hará una especial referencia a "amenazas no-maliciosas".

demasiado complejas. Algunas soluciones más recientes simplifican los métodos a expensas de volverse incompletas [Soo Hoo 00]. El presente trabajo no tiene la pretensión de haber resuelto estos problemas, pero se piensa que esta solución del diseño *intrusion-aware* propone un camino para administrar el problema del análisis de riesgo desde la perspectiva de la supervivencia. ***Si bien los trabajos mencionados anteriormente contribuyen al desarrollo de un modelo para IAD, ninguno de ellos aprovecha todo el potencial que posee la información de ataque disponible de explotar para mejorar la supervivencia del sistema. Nadie que conozcamos está analizando con profundidad el problema de utilizar los patrones y tendencias de ataques durante el refinamiento de la arquitectura del sistema para mantener la seguridad y la supervivencia del sistema, de tal manera que hacer frente de una manera adecuada a la naturaleza cambiante del ambiente de amenaza.*** El objetivo de este trabajo es resolver este problema, tratando directamente con el mantenimiento de la supervivencia a media que tanto la misión y la arquitectura del sistema como el ambiente de amenazas cambien. ***TRIAD está orientado a mitigar el riesgo de supervivencia a un nivel de la arquitectura.*** *No pretende "reinventar"el análisis de riesgo de la seguridad, pero propone ciertas técnicas de análisis como apropiadas. En el largo plazo, se pretende mejorar la precisión y la rapidez de las técnicas de análisis de riesgo mediante la documentación de los patrones de ataques más comunes que se repiten periódicamente en un formato genérico y reutilizable.*

4.2 VISION GENERAL DEL MODELO TRIAD

Está ampliamente aceptado que buena para de la arquitectura de un sistema resulta de naturaleza creativa:

> *"Los procesos de diseño de la arquitectura son inherentemente eclécticos y muy variados, pasando abruptamente de ser intensamente creativos e individualistas a muy prescriptos y rutinarios. A pesar que los procesos pueden ser eclécticos, también se los puede organizar. De los diferentes conceptos de organización, uno de los más útiles es la progresión etapa por etapa o "refinamiento"* [Maier 00].

El modelo TRIAD se formuló en torno a la noción central de refinamiento del diseño de la arquitectura, del cual deriva la "R" del TRIAD. Esta sección describe una visión general de la estructura del modelo, seguida por una discusión más detallada de la ejecución del modelo para producir una estrategia de supervivencia robusta y una implementación técnica de esa estrategia.

4.2.1 Estructura del Modelo

Maier observa que el proceso de definir la arquitectura de un sistema puede ser bien "caracteri-

zado como episódico, con episodios de reducción de la abstracción que alternan con episodios de reflexión y expansión del objetivo" [Maier 00]. A los fines de reflejar esta naturaleza episódica, TRIAD adopta la estructura y la filosofía del modelo en espiral para el desarrollo de sistemas [Boehm 88, Marmor-Squires 89]. El mencionado modelo está propuesto para el desarrollo y mejoramiento de sistemas y de software en dominios complejos en los cuales los desarrolladores poseen experiencia limitada o dominios en los que la mejor dirección (o al menos una buena) para el refinamiento del sistema resulta altamente incierta. Estos dominios requieren de un refinamiento iterativo, en el cual cada iteración refina gradualmente los requerimientos, el diseño y la implementación del sistema en base a la experiencia de cualquier iteración previa. Esta iteración permite realizar ajustes y correcciones en las direcciones elegidas para el refinamiento del sistema en base a nuevas evidencias, tales como análisis de riesgo, prototipos y simulación. El modelo en espiral prosigue a través de cuatro cuadrantes, cada uno de los cuales ejerce un progreso hacia una mejor comprensión y documentación refinada de los requerimientos, diseño y/o implementación del sistema.

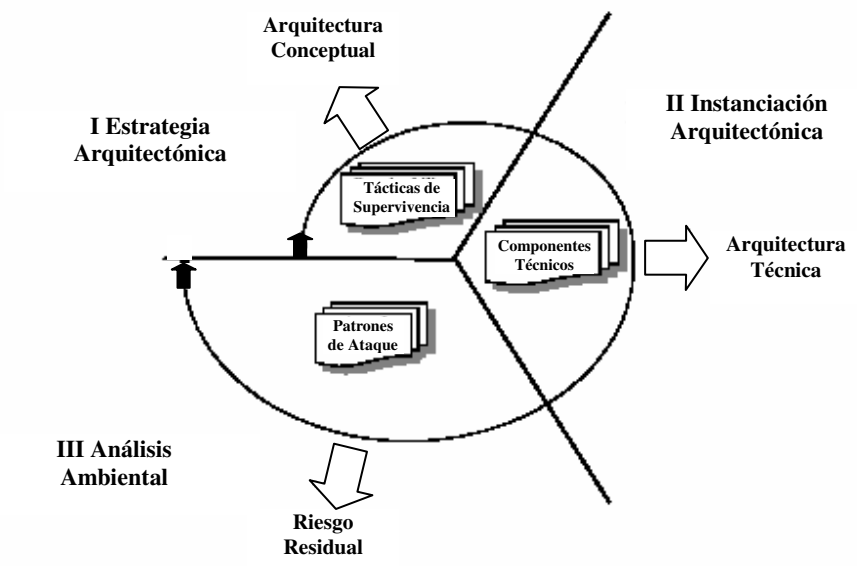


Figura 15. Reseña del proceso TRIAD

El desarrollo de sistemas de supervivencia resulta, por cierto, ser un dominio en el cual el refinamiento óptimo de la estrategia no es claro durante las primeras etapas del diseño del sistema, particularmente cuando se encuentran involucrados sistemas basados en red ilimitados. Se requiere de mucha experimentación y análisis antes de poder encontrar una solución con un grado aceptablemente bajo de riesgo residual de falla en la misión.

La estructura en espiral de TRIAD, que se muestra en la Figura 15, prosigue a través de *tres sectores*: *(I) Estrategia Arquitectónica, (II) Instanciación Arquitectónica y (III) Análisis Am-*

biental. Si bien la figura sólo muestra la estructura general del modelo, *un modelo totalmente instanciado comprende múltiples iteraciones a través de estos sectores*. Consistente con el modelo en espiral original, *cada iteración refina gradualmente la arquitectura del sistema en base a la elaboración de prototipos, análisis del riesgo y mitigación del riesgo de cualquier iteración anterior*. El proceso comienza en el medio de la figura, en el sector I, y continúa a lo largo de la espiral, en donde la dimensión angular indica el progreso acumulativo. Una instanciación del modelo implica múltiples iteraciones a través de los sectores, lo que permite la realización de ajustes y correcciones en los requerimientos, arquitectura o los riesgos resultantes en base a la nueva experiencia y evidencia. Al igual que con el modelo en espiral, TRIAD resulta aplicable al desarrollo de sistemas nuevos y al mejoramiento de sistemas existentes. La discusión que sigue describe las principales actividades dentro de cada sector.

4.2.1.1 Visión general de los Sectores

El **sector I Estrategia Arquitectónica** *comienza con la elaboración de los aspectos relacionados con la misión del sistema que se está diseñando. Las actividades dentro de este sector obtienen como resultado los requerimientos justificables de supervivencia del sistema y una arquitectura de supervivencia de alto nivel a partir de la necesidad de garantizar el éxito de la misión a pesar de penetraciones y compromisos. La arquitectura de supervivencia conceptual (a partir de aquí referida en forma abreviada como arquitectura conceptual) describe la función y la estructura del sistema a un nivel apropiado para el cliente. Como se muestra en la Figura 15, la arquitectura conceptual deriva de un conjunto de tácticas de supervivencia. Una táctica de supervivencia es una representación genérica de una solución arquitectónica para resistir, reconocer, a partir de la cual recuperarse, o adaptarse frente a un ataque dentro de un contexto específico*[14]*. Las tácticas de supervivencia describen las respuestas estratégicas a los patrones generales de un ataque*, tales como las diferentes formas de ataque de denegación de servicio y de respuesta frente al mismo [CERT 01]. ***Una arquitectura conceptual creada a partir de estas tácticas de supervivencia constituye la estrategia de supervivencia para el sistema.***

Las actividades dentro del **sector II Instanciación Arquitectónica** *refina la arquitectura técnica dentro del conjunto de restricciones establecidas por la arquitectura conceptual mediante la identificación e integración de los bloques constructivos técnicos críticos. La arquitectura de supervivencia técnica* (de aquí en más abreviada como arquitectura técnica) *describe la función y la estructura del sistema en un nivel de detalle técnico suficiente como para construir el sistema. Las* ***actividades del sector continúan con la identificación de los componentes técnicos de bajo nivel para instanciar la arquitectura conceptual****.* Un componente técnico es cualquier bloque construc-

[14] El uso del término táctica dentro de este contexto no implica que el enfoque arquitectónico sólo soluciona objetivos de corto plazo, sino que dicho enfoque soluciona aspectos específicos dentro de un contexto aislado.

tivo arquitectónico tal como un hardware o un software comercial de tipo COTS.

Las actividades dentro del **Sector III Análisis Ambiental** *representan el ambiente de amenazas y analiza su impacto en la operación del sistema, incluyendo la aptitud del sistema para realizar exitosamente su misión crítica.* ***El ambiente de amenazas se deriva de un conjunto de patrones de ataque.*** *Un patrón de ataque es una representación genérica de una actividad deliberada y maliciosa que comúnmente ocurre dentro de un contexto arquitectónico específico. Una patrón de ataque puede tener por objetivo personas* (por ejemplo, ataques de ingeniería social que utilizan un virus de computadora), *la operación de la tecnología* (por ejemplo, ataques de denegación de servicio distribuido), *o el contexto en el cual las personas deben trabajar* (por ejemplo, ataques *dumpster diving*).

Las diferencias entre los sectores no siempre resultan claras, y generalmente existe cierto solapamiento, como sucedía en el modelo de espiral original. Sin embargo, tenemos diferencias bastante concretas entre cada uno de los tres sectores. La diferencia entre el Sector I y el II es similar a la diferencia entre requerimientos y una especificación. Los requerimientos pueden describir una estrategia de solución general, pero dejan abiertos muchos detalles a nivel de diseño e implementación; una especificación toma muchas decisiones concretas sobre cómo proceder, a menudo en términos de componentes y conectores específicos. Las actividades del Sector I refinan de modo *top-down* la arquitectura conceptual a partir de los objetivos de misión, mientras que las actividades del Sector II instancian la arquitectura conceptual, como una arquitectura técnica, a partir de los componentes técnicos disponibles. Tanto refinamiento como instanciación son una parte esencial en el proceso de desarrollo de un sistemas, y TRIAD los soporta explícitamente en cada iteración. La combinación de la arquitectura conceptual y la arquitectura técnica conforman la arquitectura de supervivencia de los sistemas. Finalmente, el Sector III se centra en el análisis de amenazas y el impacto dadas las restricciones arquitectónicas especificadas en el Sector II, mientras que el Sector I se centra en la descripción de los requerimientos para mitigar el riesgo resultante. Las actividades del Sector III aseguran que el ambiente de amenazas sea considerado de manera consistente a lo largo de todas las iteraciones del refinamiento arquitectónico.

4.2.1.2 Relaciones entre los datos

La relación esencial entre los datos sobre los cuales se basa cada Sector se muestra en la Fig. 16. Los enfoques estratégicos que aseguran el éxito de la misión sugieren el empleo de componentes técnicos específicos para la supervivencia. Estos componentes técnicos, a su vez, poseen ciertas vulnerabilidades dentro del contexto de una arquitectura de sistema que fomenta ciertos patrones de ataque. Éstos, a su vez, sugieren la adopción de otras tácticas de supervivencia. De

hecho, esto podría conducir a un ciclo de análisis que no tiene fin. *El reto para el diseñador de intrusion-aware es converger hacia un conjunto de tácticas de supervivencia, cada miembro de las cuales está implementado como un conjunto de componentes técnicos, que muy probablemente solucionan patrones de ataque de una manera accesible y efectiva.*

Puede resultar necesario el mantenimiento del diseño debido a cambios en los objetivos de la misión, cambios en la arquitectura subyacente, o cambios en el ambiente de amenazas.

- *La misión del sistema puede sufrir una expansión o contracción de su misión*. La contracción de la misión o la modificación de su naturaleza puede, por ejemplo, requerir centrarse en la transición de un *e-business* orientado a la venta de mercaderías prestigiosas hacia una estrategia de ventas de alto volumen de productos discontinuados de mercaderías baratas debido a diferentes presiones del mercado.

- *Los cambios en la arquitectura del sistema pueden ser de naturaleza procedural o tecnológica*. Un negocio puede decidir relajar las prácticas de empleo en respuesta a un mercado de trabajo altamente competitivo. Un cambio tecnológico puede producirse cuando un *e-business* expande físicamente los puntos de ventas en múltiples sitios distribuidos; este cambio podría requerir de herramientas de administración de inventario y un nivel de confianza en los *workflows* entre los sitios distribuidos.

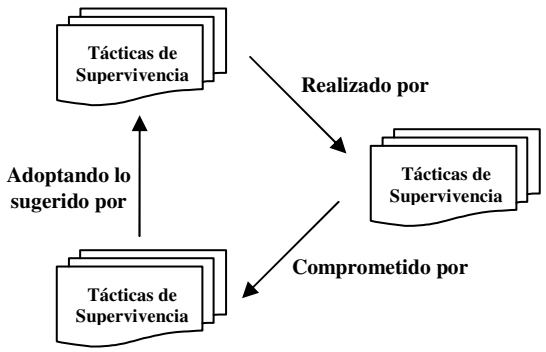


Figura 16. Relaciones entre datos

- *Los cambios en el ambiente de amenazas puede involucrar a nuevos tipos de atacantes que se necesitan tener en cuenta, o a viejos tipos de atacantes que utilizan nuevos métodos*. Los nuevos tipos pueden volverse contra un *e-business* cuando, por ejemplo, los reportes recientes hacen publicidad sobre sus tratos con organizaciones impopulares, haciendo que sus operaciones de sistemas sean más susceptibles de ataques "activistas"; los atacantes que ya fueran considerados como una amenaza podrían volver a tener relevancia con la aparición de nuevas herramientas de ataque que se pueden utilizar para penetrar el perímetro de la corporación y tomar el control de las operaciones de la Intranet.

TRIAD enfatiza las interrelaciones entre los tres sectores. Los cambios en los objetivos de la misión pueden conducir directamente a cambios en la estructura del sistema para soportar los objetivos modificados. A su vez, los cambios en la estructura del sistema afectan al ambiente de amenazas, por ejemplo, mediante el incremento de la exposición. Finalmente, un cambio en el ambiente de amenaza puede conducir a modificar los requerimientos para preservar la supervivencia, y finalmente, los cambios estructurales para soportar estos requerimientos.

La documentación alentada por el modelo pone énfasis en la rastreabilidad entre los artefactos del sector para soportar el mantenimiento continuo de la supervivencia del sistema aún luego que el sistema haya sido creado.

4.2.2 Ejecución del Modelo

La ejecución de TRIAD, visto desde un nivel abstracto, comienza con el desarrollo de la estrategia de supervivencia y continúa con la implementación de esa estrategia como una arquitectura de supervivencia concreta y específica. La Figura 17 muestra que *las iteraciones iniciales del modelo se centran en la definición de la estrategia de supervivencia. Las iteraciones posteriores se centran en el refinamiento técnico de esta estrategia.*

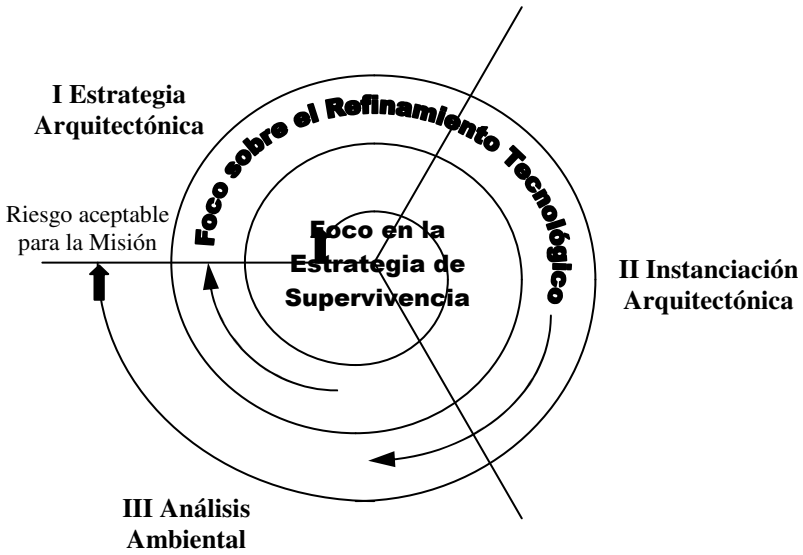


Figura 17. Ejecución de TRIAD

En tanto el centro inicial se ubica en la composición de las tácticas de supervivencia a los fines de producir la estrategia de supervivencia, el desarrollo de dicha estrategia no puede estar aislado de su refinamiento técnico. Este refinamiento requiere de un cierto análisis de factibilidad técnica para asegurar que la estrategia es, en realidad, implementable de una manera efectiva respecto al costo. De la misma manera, el refinamiento de la arquitectura técnica requiere de un análisis top-down para asegurar que la estrategia está siendo implementada en respuesta

a la misión global. El avance continúa hasta que queda definitivo el conjunto de artefactos producidos por cada sector, y el nivel de riesgo residual de falla de la misión determinado por las actividades del Sector de Análisis Ambiental resulta aceptable para los actores involucrados. El número exacto de iteraciones de la espiral requeridas hasta su finalización varía dependiendo de la complejidad de la aplicación y de la experiencia en el dominio de aplicación de los desarrolladores, pero por lo general se espera su convergencia en una solución aceptable dentro de las dos a cuatro iteraciones.

A continuación se proporciona una reseña del desarrollo de la estrategia de seguridad que toma cuerpo como una arquitectura conceptual, seguida de una reseña del refinamiento técnico de la estrategia dentro de una arquitectura técnica. La aproximación descripta aquí ayuda a comprender el impacto de un ambiente de amenazas maliciosas sobre la aptitud de la organización en cumplir su misión a todos los niveles de responsables de toma de decisiones. Además, la aproximación ayuda a formular una respuesta estratégica accesible y efectiva a los ataques que probablemente han de comprometer la misión. Si ya está planificada o definida una respuesta, la aproximación ayuda a evaluar la respuesta a la luz de la amenaza y hacer recomendaciones para mejorar su eficacia. La aproximación facilita la planificación de los inevitables cambios en el ambiente de amenazas y en las operaciones del sistema, y a determinar el efecto de dichos cambios sobre el éxito continuo de la misión.

4.2.2.1 Focalización en la Estrategia de Supervivencia

Una organización posee una misión implícita o explícita que caracteriza su propósito principal como un conjunto de objetivos de alto nivel. La tecnología de información, las políticas, los procedimientos, el personal, y el contexto de trabajo global existen para sostener a la misión. Debemos ser capaces de evaluar en cualquier etapa del refinamiento arquitectónica el impacto del posible desarrollo de un ambiente de amenazas sobre el sistema y su misión global como fuera descripto. La manera en que el método aborda el desarrollo y la evaluación de la estrategia de supervivencia se basa en una rama de la investigación de operaciones llamada ***dinámicas del sistema*** (*system dynamics*) [Sterman 00], habiéndose elaborado un sub-dominio especializado que se denominó ***dinámicas de amenaza*** (*thread dynamics*), que interpreta la dinámica del sistema para incluir las acciones hostiles en forma explícita y la respuesta operativa del sistema a tales acciones. *Las dinámicas de amenaza permiten el modelado de la estructura y las dinámicas de sistemas complejos basados en el comportamiento humano, de entre los cuales la relación entre la comunidad de atacantes basados en Internet y los sistemas de información basados en Internet es un ejemplo específico. Mediante la definición de una visión holística del ambiente de amenaza dentro del contexto de las operaciones del sistema existente o propuesta,*

las dinámicas de amenaza proveen una reseña de las influencias generales que el ambiente de amenaza tiene sobre la aptitud del sistema para satisfacer su misión y una mejor comprensión de las respuestas estratégicas para contrarrestar las posibles amenazas.

La Figura 18 describe la aproximación a alto nivel para el refinamiento de la estrategia de supervivencia. La estrategia, la cual toma cuerpo como una arquitectura conceptual, se deriva iterativamente a través de su evaluación utilizando las dinámicas de amenaza. La estrategia deriva de los objetivos globales de la misión y de la experiencia que se posee con patrones recurrentes de ataques de alto nivel que representan los principales escenarios de intrusión que se deben considerar. Las tácticas de supervivencia son aproximaciones arquitectónicas amplias que aseguran que tales ataques no se ciernen sobre la supervivencia de la misión. Dichas tácticas pueden ayudar a formular la arquitectura conceptual, pero tal formulación debe resultar apropiada respecto de las restricciones operacionales del dominio de aplicación.

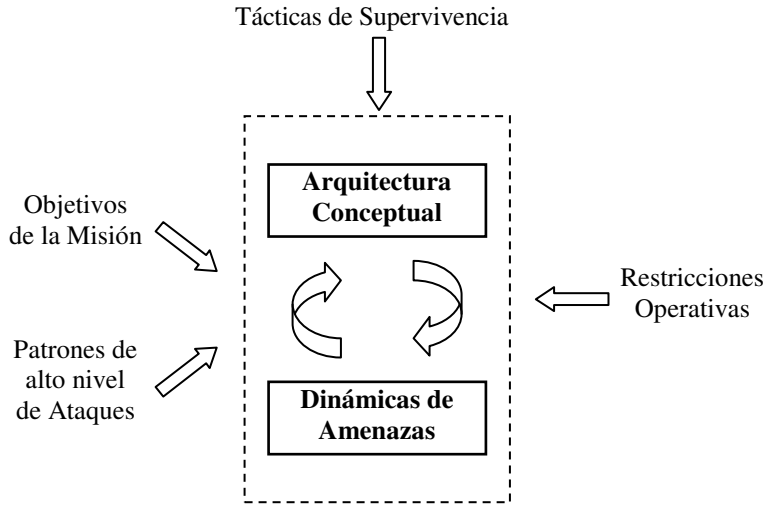


Figura 18. Reseña del proceso de refinamiento de la estrategia de supervivencia.

En término de los sectores TRIAD, la arquitectura conceptual se define principalmente en el Sector I y se la evalúa utilizando las dinámicas de amenazas en el Sector III. Las restricciones operacionales surgen principalmente debido a consideraciones técnicas del Sector II. La mayor parte del desarrollo y la evaluación de la estrategia de supervivencia tienen lugar en los Sectores I y III. Un rol esencial para el Sector II es el de asegurar que la estrategia de supervivencia se puede implementar, si bien los detalles de exactamente cómo hacerlo se dejan para un refinamiento técnico posterior. Además, puede resultar necesario ajustar la arquitectura conceptual a los fines de satisfacer restricciones técnicas que no fueron previamente consideradas. *En tanto las dinámicas de amenaza ayudan a la comprensión de la influencia del ambiente de amenaza sobre la aptitud de cumplir la misión, la rastreabilidad de la supervivencia ayuda a documentar y justificar el soporte que proporciona la arquitectura de supervivencia a la misión. La rastreabilidad de la supervi-*

vencia resulta esencial para la administración de los cambios de tal manera que mantiene la supervivencia de la organización a lo largo del tiempo. En un contexto más amplio, la rastreabilidad se puede definir como "una característica de un sistema en el cual los requerimientos están claramente vinculados a sus orígenes (rastreabilidad hacia atrás) y a los artefactos creados durante el ciclo de vida de desarrollo del sistemas en base a estos requerimientos (rastreabilidad hacia adelante)" [Ramesh 97]. En esta definición, se considera que los vínculos son bidireccionales. El Sector I de TRIAD es responsable de la rastreabilidad hacia atrás con respecto a los objetivos de la misión, mientras que el Sector II es responsable de la rastreabilidad hacia delante con respecto a la arquitectura técnica. Las amenazas atendidas son las identificadas en el Sector III.

La rastreabilidad de los requerimientos y de las alternativas de decisión a partir de la misión de una organización ayuda a determinar las dependencias de supervivencia de un sistema. La rastreabilidad hacia atrás puede ayudar a evaluar el impacto de los cambios en la misión o del ambiente de amenazas de una organización. La rastreabilidad hacia delante puede ayudar a evaluar el impacto de los cambios en la arquitectura del sistema. Un criterio aceptado por la comunidad de rastreabilidad de requerimientos estipula que *sólo debe ser mantenida la rastreabilidad de los requerimientos críticos para la misión* [Ramesh 98]. *Este criterio está exactamente alineado con el hecho de poner a la misión como centro de la supervivencia, dado que la misión de la organización proporciona el punto de partida para la rastreabilidad del método TRIAD.*

La rastreabilidad de la supervivencia requiere de un alcance más amplio del que generalmente se adopta para la rastreabilidad de los requerimientos debido a la variedad de las amenazas (es decir, desde la ingeniería social hasta el compromiso tecnológico) y de las medidas en contra (es decir, desde el personal, pasando por lo procedural y hasta lo tecnológico).

La elección de los requerimientos y del diseño debe ser administrada de manera consistente durante todo el tiempo de vida del sistema para soportar la continua administración del riesgo, de tal manera que las nuevas amenazas y operaciones del sistema no conduzcan a la falla de la misión. Dado que TRIAD es un proceso de refinamiento iterativo, la misión, las amenazas, los requerimientos y la arquitectura pueden estar sólo parcialmente definidas en cualquiera de las iteraciones de la espiral. En consecuencia, de la misma manera, la definición de los requerimientos y sus rastros se produce en forma incremental.

4.2.2.2 Focalización en la Estrategia de Refinamiento

La implementación de la estrategia de supervivencia comprende el desarrollo de una arquitectura técnica que instancia la arquitectura conceptual. La Figura 19 describe muestra este enfoque a alto nivel del refinamiento de la arquitectura técnica dentro de las restricciones impuestas por la arquitectura conceptual. ***La arquitectura técnica se deriva iterativamente a través de su evalua-***

ción utilizando una técnica denominada árboles de ataque (*attack trees). Los árboles de ataque se pueden construir mediante la composición de los patrones de ataque de bajo nivel que han probado ser probables y críticos. Los patrones de ataque de interés para una aplicación particular son aquéllos que pueden comprometer la misión. El refinamiento técnico debe mantener la rastreabilidad de la arquitectura conceptual a lo largo de toda la arquitectura técnica.*

Las áreas críticas de consideración en el desarrollo de la arquitectura técnica incluyen:

- La Intranet, la cual incluye las bases de datos, las aplicaciones, los servidores, las estaciones de trabajo, las redes internas de la organización y los procedimientos utilizados por todos ellos.
- El Perímetro, el cual incluye los *firewalls*, los *gateways* y los mecanismos físicos destinados a proteger a los activos en la Intranet de la organización contra el acceso externo.
- La Extranet, la cual incluye cualquier red que se encuentre fuera del Perímetro y con la que se cuenta para alcanzar la misión de la organización

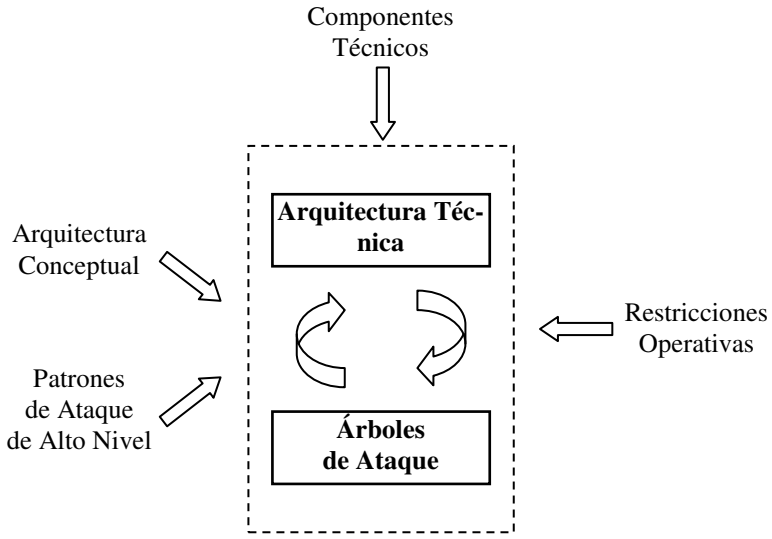


Figura 19. Reseñan del proceso de refinamiento de la arquitectura técnica.

Las tácticas de supervivencia que se utilizan en la arquitectura conceptual se implementan en términos de componentes técnicos disponibles (ítems que no son objeto de desarrollo) y, cuando resulte necesario, de componentes desarrollados a medida. Los componentes técnicos pueden estar disponibles ya sea comercialmente o a través de programas de investigación y desarrollo. Los refinamientos técnicos comprenden la identificación de las responsabilidades de los componentes individuales de la arquitectura técnica que ayudan a alcanzar los requerimientos de supervivencia. Los requerimientos de refinamiento pueden demandar una negociación en términos de costos o de complejidad de la administración, la cual puede exigir cambios en la arquitectura conceptual.

El extenso número de intrusiones posibles para cualquier sistema no-trivial necesita de un esquema para organizar las intrusiones relacionadas. Los árboles de ataque proveen un esquema organizacional de este tipo [Salter 98, Scheneier 99, Schneier 00a]. Los mismos refinan la información acerca de las intrusiones mediante la identificación del compromiso de la seguridad o la supervivencia de la organización como la raíz del árbol. Las maneras en las que un atacante puede causar este compromiso se refinan incrementalmente como los nodos de bajo nivel del árbol. Generalmente, un sistema posee un conjunto, o bosque, de árboles de ataque que son relevante para su operación. La raíz de cada árbol dentro de un bosque representa un evento que podría dañar significativamente la misión del sistema. Cada árbol de ataque enumera y elabora las maneras en las que un atacante podría hacer que ocurra un evento. Cada camino a través del árbol de ataque representa una intrusión única sobre la organización. Un nodo de un árbol de ataque se descompone como uno de los siguientes:

- Un conjunto de sub-objetivos de ataque que están representados como una descomposición AND. Todos estos objetivos deben ser alcanzados para que el ataque tenga éxito.
- Un conjunto de sub-objetivos de ataque que están representados como una descomposición OR. Si se alcanza cualquier de estos objetivos, el ataque tiene éxito.

Representamos gráficamente las descomposiciones como se muestra en la Figura 20. La descomposición AND representa un objetivo G_0 que se puede alcanzar en caso que el atacante logre todos los objetivos, desde G_1 hasta G_n. La descomposición OR representa un objetivo G_0 que se puede alcanzar en caso que el atacante alcance uno cualquiera de los objetivos comprendidos entre G_1 y G_n. En la práctica, a menudo los árboles de ataque se representan en forma de texto, dado que la representación gráfica por ser dificultosa para árboles de ataque no-triviales.

Los árboles de ataque constan de cualquier combinación de descomposiciones AND y OR. Se generan escenarios de intrusión particulares a partir de un árbol de ataque recorriéndolo en profundidad; un ejemplo de esto se presenta en la Figura 20. En general, las hojas de objetivos se agregan sobre el final de los escenarios de intrusión a medida que los mismos son generados. Las descomposiciones OR causan la generación de nuevos escenarios. Las descomposiciones AND hacen que los escenarios existentes se extiendan. Los nodos intermedios del árbol de ataque no aparecen en los escenarios de intrusión, dado que los mismos son elaborados por objetivos de menor nivel.

Los árboles de ataque permiten el refinamiento de los ataques a un nivel de detalle elegido por el desarrollador. Los mismos exhiben la propiedad de transparencia referencial como fuera caracterizada por Prowell:

> *"La transparencia referencial implica que los detalles de menor nivel de una entidad*

> *relevante son abstraídos, en lugar de omitidos, dentro de un nivel particular mediante una descripción de nivel superior, por lo que esta descripción contiene todo lo necesario para comprender a la entidad cuando se la ubica en un contexto más amplio"* [Prowell 99]

Esta propiedad permite que el desarrollador explore ciertos caminos de ataque en mayor profundidad que otros, y aún así, que los escenarios de intrusión generados tengan sentido. Además, el refinamiento de las ramas del árbol de ataque genera nuevas hojas, dando por resultado escenarios de intrusión a un nuevo nivel más bajo de abstracción. La noción de transparencia referencial es crítica para poder administrar la complejidad inherente de las representaciones del árbol de ataque mediante la restricción del refinamiento a un nivel arquitectónico de abstracción. Moore describe los detalles de la aproximación anterior, ofreciendo un ejemplo de su aplicación [Moore 01a].

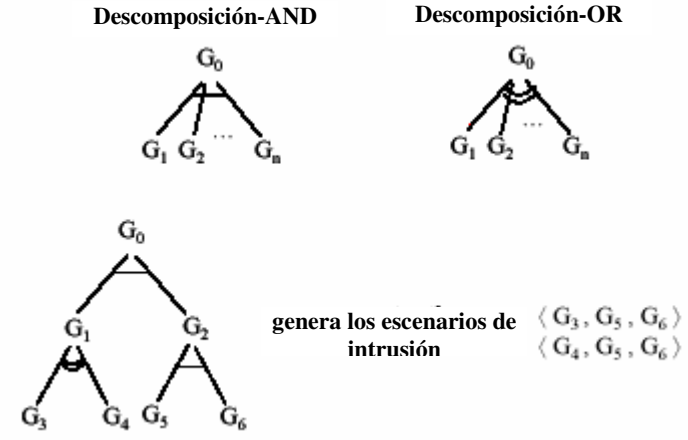


Figura 20. Representación del árbol de ataque.

Los árboles de ataque se pueden utilizar para mejora una arquitectura técnica mediante el planteo de preguntas referidas a la resistencia, reconocimiento, recuperación y adaptación en cada uno de los nodos del árbol de ataque. Las preguntas acerca de la resistencia plantean cómo evitar que un atacante tenga éxito en su intento de atravesar este nodo para poder comprometer la misión; desde luego que las respuestas no siempre dan por resultado una solución práctica o efectiva en cuanto al costo. Lo que resulta fundamental para el objetivo de la supervivencia es la existencia de planes de recuperación para aquellos ataques que no pueden ser efectivamente evitados. En consecuencia, nos preguntamos:

- ¿Cómo podemos detectar un atacante durante un ataque frustrado o luego de un ataque exitoso?
- ¿Cómo podemos lograr la recuperación a partir de algún compromiso?
- ¿Cómo podemos adaptar el sistema de tal manera que esa intrusión no pueda volver a suceder?

Resistir una rama OR de un árbol de ataque significa resistir los escenarios de intrusión asociados con esa rama. Los nodos de resistencia al ataque ubicados en la parte más alta de la jerarquía del árbol de ataque proporcionan un bloqueo del atacante más efectivo, pero asimismo los cambios en la arquitectura del sistema y las operaciones son, potencialmente, más costoso y de mayor alcance. *Resistir una rama AND de un árbol de ataque significar resistir todos los escenarios de intrusión asociados con el nodo padre de la rama*. Este efecto de palanca se logra debido a que el atacante debe atravesar todas las ramas de una descomposición AND para lograr su objetivo; resistir una cualquiera de las ramas AND resiste el objetivo definido por el nodo padre. La mejor técnica (o combinación de técnicas) se elige en base al costo, la factibilidad y la certidumbre de la implementación. *El tipo de reconocimiento, recuperación y adaptación necesarios depende del tipo de ataque, es decir, de la rama del árbol de ataque atravesada.* Los árboles de ataque sólo necesitan estar definidos en un nivel que soporte el análisis arquitectónico de tal manera que los responsables a nivel empresa puedan aceptarlo como suficiente y accesible. Moore describe un ejemplo simple de este tipo de análisis [Moore 01b].

4.2.3 *Intrusion-Awareness* en TRIAD

La figura 21 muestra ***una visión de conjunto del análisis arquitectónico TRIAD referido a la robustez ante los ataques que comprometen la misión***. *El tipo de análisis varía desde lo puramente estratégico, en la parte superior de la figura, hasta lo puramente práctico, en la parte inferior de la misma. Las dinámicas de ataque se utilizan para modelar el impacto estratégico del ambiente de amenazas sobre la aptitud de la organización en lograr su misión, y para determinar las respuestas estratégicas que atenúan los efectos adversos. Las dinámicas de ataque permiten adentrarse y comprender las alternativas posibles de las estructuras y estrategias de negocio que sacan provecho de manera óptima de la tecnología de la información y clarifican el rol de esa tecnología para asegurar la supervivencia de la misión de la organización.*

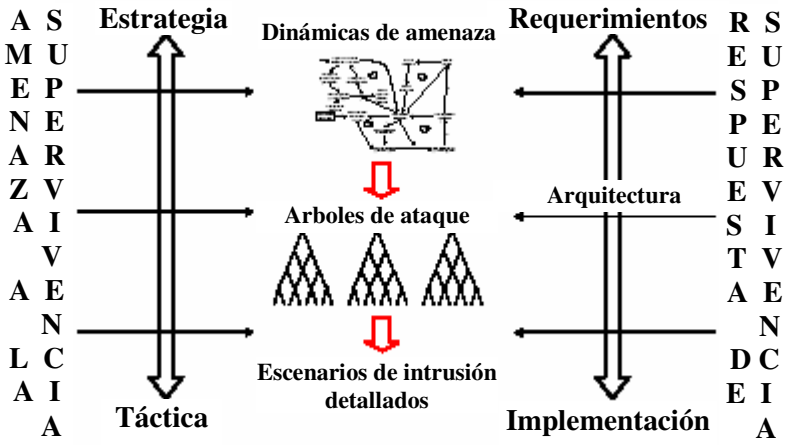


Figura 21. Análisis estructura de intrusión

La solución descripta administra la complejidad del problema del análisis del riesgo de supervivencia centrándose sólo en las amenazas que pueden comprometer la misión y sólo en las vulnerabilidades a un nivel arquitectónico grueso. La naturaleza holística del punto de arranque de las dinámicas de amenaza ayuda a asegurar que todas las potenciales áreas de amenazas y soluciones hayan sido consideradas hasta un nivel de análisis arquitectónico. Las dinámicas de amenaza proporcionan un medio para analizar los efectos de las tendencias observadas en el comportamiento del atacante. La relación entre las amenazas y las formas de mitigar el riesgo se preserva a lo largo de la trazabilidad de la supervivencia. El análisis de las dinámicas de amenaza se ve beneficiado con la disponibilidad de datos referidos al incidente y la vulnerabilidad en caso que éstos existan, pero no es dependiente de dichos datos en lo referente a proporcionar un entendimiento útil del impacto del ambiente de amenazas sobre las operaciones de los sistemas.

El resto del trabajo se centra en los aspectos estratégicos de TRIAD, incluidos los aspectos de dinámicas de amenaza de la Figura 21. Como ya se mencionara anteriormente, muchas de las metodologías de desarrollo de un sistema seguro y de las técnicas de análisis de riesgo existentes en la actualidad se centran más en los aspectos técnicos del análisis de la amenaza y la respuesta asociada. Además, el trabajo previo en el CERT/CC ha delineado un enfoque para el empleo de los árboles de ataque en el análisis de supervivencia de sistemas en un nivel técnico [Moore 01a]. En este sentido, *los métodos identificados y ejemplificados en el presente trabajo llenan un vacío de carácter crítico en la habilidad de construir sistemas de información seguros y con supervivencia*.

4.3 DOCUMENTACION DE LA ESTRATEGIA DE SUPERVIVENCIA

En el contexto de TRIAD, ***una estrategia de supervivencia es una aproximación integrada a partir de la cual resistir, reconocer, recuperarse y adaptarse ante ataques que comprometen la misión***.

Los objetivos de la estrategia de supervivencia son:

- ***Proveer una respuesta documentada frente a las amenazas principales a la misión***. Ejemplos de esto incluyen la variedad de respuestas posibles ante ataques de denegación de servicio [CERT 01]. La estrategia debería documentar la respuesta y el entrenamiento esperados tanto de parte de la administración del sistema como de las operaciones en soporte de esa respuesta.
- ***Proveer una justificación del diseño del sistema y las limitaciones***. La estrategia proporciona un diseño racional que describe de qué manera la arquitectura da sustento a la

respuesta esperada ante las amenazas. La justificación puede incluir la política de aseguramiento de la información, los requerimientos de certificación, o los argumentos pertinentes en la toma de decisiones. Los documentos de estrategia diseñan las opciones y los balances, y proporcionan el *input* para la revisión, supervisión y testeo del diseño y la implementación. A menudo las limitaciones se expresan como supuestos del diseño que se deben validar para sostener la justificación.

- ***Sustentar el diseño y la implementación del comportamiento deseado del sistema a través de múltiples sistemas y múltiples equipos de desarrollo***. La estrategia se encuentra documentada de tal manera que permite la comunicación entre múltiples equipos de desarrollo durante las etapas de adquisición e ingeniería. Durante la adquisición, la estrategia provee un excelente punto de partida para solicitar propuestas, y puede resultar de utilidad para evaluar las respuestas a dichas propuestas. La estrategia documenta los riesgos y responsabilidades compartidos entre múltiples organizaciones. Además, da sustento a la incorporación de requerimientos de seguridad del sistema en el proceso de desarrollo de software tal como se describe en [DoD 02, DISTCAP 99].
- ***Sustentar el mantenimiento y la evolución del sistema***. La estrategia ayuda al mantenimiento de los supuestos de diseño y a la verificación de la continuidad en la eficiencia de la respuesta a las amenazas. Esto puede abarcar el análisis del impacto de nuevas amenazas y los cambios en el ambiente de operativo. La estrategia utilizada por un sistema existente puede tener que pasar por un proceso de re-ingeniería a partir de la documentación existente en caso que la estrategia no hubiera sido documentada explícitamente como parte del proceso de desarrollo global del sistema.

Los principales artefactos de información relacionados con la estrategia de supervivencia incluyen:

- ***Objetivos de la misión***. El propósito de alto nivel del sistema desde el punto de vista de los propietarios del sistema.
- ***Amenazas sobre la misión***. Las amenazas en alcanzar los objetivos de la misión.
- ***Requerimientos de supervivencia***. Los requerimientos que dan soporte a la resistencia, reconocimiento, recuperación a partir de, y adaptación a las amenazas sobre la misión.
- ***Arquitectura conceptual***. Una descripción de la estructura y función del sistema que garantiza que los requerimientos de supervivencia sean atendidos con suficiente certeza.

Los requerimientos de supervivencia y la arquitectura conceptual consideradas en conjunto constituyen la estrategia de supervivencia. A continuación se caracterizan con mayor detalle estos artefactos, comenzando con la trazabilidad requerida entre ellos.

4.3.1 Trazabilidad de la Supervivencia

La justificación de la estrategia de supervivencia requiere del argumento de que la arquitectura conceptual soporta el éxito de la misión. Desde hace mucho tiempo la trazabilidad ha estado siendo utilizada como ayuda para garantizar que el diseño y la implementación de un sistema responden a sus requerimientos [Ramesh 97]. ***Se define la trazabilidad de la supervivencia como la característica de un sistema en el cual los requerimientos de supervivencia están claramente vinculados con sus fuentes (objetivos de la misión) y con los artefactos creados durante el ciclo de vida de desarrollo del sistema basado en estos requerimientos (arquitectura de supervivencia)***.

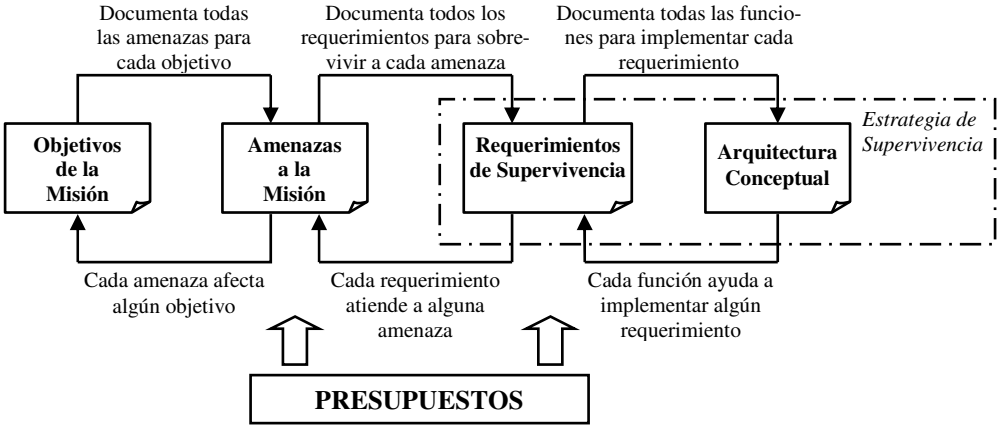


Figura 22. Trazabilidad de supervivencia desde la emisión hasta la arquitectura conceptual.

La Figura 22 ilustra la trazabilidad de los objetivos de la misión comenzando por las amenazas de la misión y continuando hasta los requerimientos de supervivencia, incluidos tanto los requerimientos funcionales como los no-funcionales del sistema. Luego, se trazan los requerimientos de supervivencia respecto a la arquitectura conceptual. Justificar la estrategia de supervivencia requiere argumentar que la arquitectura conceptual soporta el éxito de la misión. Este argumento depende de la trazabilidad de la supervivencia como un todo y probablemente dependa de un conjunto de supuestos acerca del ambiente operacional, por ejemplo, la estabilidad de las interfaces externas del sistema. Tales suposiciones deben estar documentadas como parte del proceso de trazabilidad.

La trazabilidad de la supervivencia también juega un rol esencial ayudando a los administradores de sistema a gestionar los cambios inevitables (y a menudo inesperados) de los objetivos de la organización, su estructura, comportamiento, o ambiente de amenazas de tal manera que se mantenga la supervivencia de la organización a lo largo del tiempo. Esto requiere de una gestión de cambios proactiva que ayude a determinar y, cuando sea posible, contener los efectos del cambio. El mantenimiento de la trazabilidad del sistema estimula la gestión de cambios dentro de TRIAD:

- Los efectos de los cambios en la arquitectura o en el ambiente de amenazas sobre la misión se pueden evaluar siguiendo la traza de abajo hacia arriba.
- Los efectos de los cambios en la misión o en el ambiente de amenazas sobre la arquitectura se pueden evaluar siguiendo la traza de arriba hacia abajo.

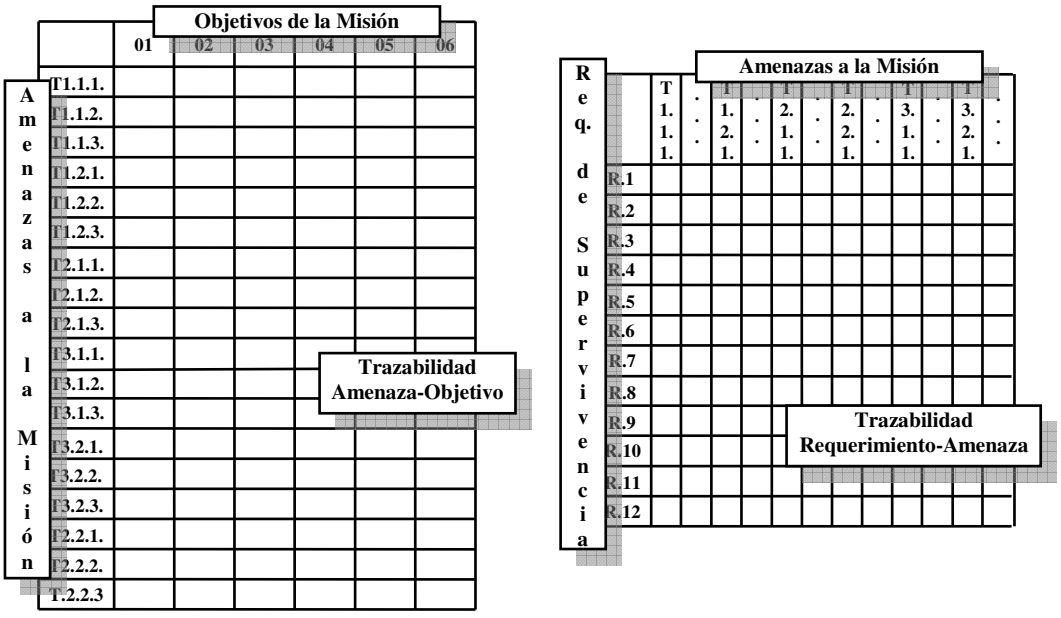


Figura 23. Ejemplo de Tablas de Supervivencia.

La aparición de nuevas amenazas o nuevas funcionalidades sustanciales dentro de la arquitectura puede exigir reingresar en el proceso de desarrollo de la supervivencia y poner al día la trazabilidad. En estos casos, la dinámica de amenaza proporcionará una valiosa asistencia para comprender el impacto de los cambios sobre la aptitud en cumplir la misión. La trazabilidad puede ser documentada en formato de tablas, como se ejemplifica en la Figura 23.

En resumen, *la trazabilidad ayuda a los encargados de la toma de decisiones a*:

- *Demostrar que los requerimientos críticos para la misión son satisfechos.*
- *Identificar la fuente y la justificación de las elecciones en cuanto a requerimientos y diseño.*
- *Comprender el impacto de los errores y las fallas en la aptitud del sistema para alcanzar su misión.*

A continuación se describe con más detalle la estructura de la documentación de artefactos de la estrategia de supervivencia: objetivos de la misión, amenazas de la misión, requerimientos de supervivencia y arquitectura conceptual.

4.3.2 Artefactos de Documentación

La estrategia de supervivencia es un trabajo que tiene lugar a lo largo del ciclo de vida del desarrollo del sistema. *Podrá comenzar con la documentación de las amenazas identificadas y su*

impacto sobre las operaciones. Una versión posterior podrá agregar la guía de diseño y las especificaciones de supervivencia tanto para la infraestructura computacional como para las aplicaciones soportadas. Próximo a producirse la implantación, se documentará las respuestas ante amenazas y sus justificaciones respaldadas en el testeo y la certificación para su aprobación.

El análisis de la supervivencia comienza con una descripción del ambiente operacional esperado y el soporte computacional deseado para los procesos esenciales, como se los podría encontrar en un nivel conceptual de las operaciones. La descripción del ambiente operacional se va refinando a lo largo del ciclo de vida del sistema. El alcance del análisis depende de la naturaleza del ambiente operacional: la complejidad de las interacciones del sistema, la distribución de los procesos, las dependencias trabajo-proceso, y la compartición de los riesgos y responsabilidades de supervivencia entre múltiples organizaciones.

La estrategia de supervivencia requiere de la comprensión de las propiedades operacionales de alto nivel (incluidas la misión y las amenazas) que afectan al análisis de supervivencia y los criterios de diseño. La estrategia de supervivencia documenta los requerimientos de supervivencia del sistema y de su arquitectura conceptual dentro de las restricciones impuestas por el ambiente operacional.

El resto de esta sección provee una reseña de los principales artefactos de documentación. La Sección 4 proporciona más información sobre estos artefactos dentro del contexto del proceso de desarrollo de la estrategia de supervivencia.

4.3.2.1 Objetivos de la misión

La supervivencia requiere que los objetivos de la misión se encuentren explícitamente documentados y que el logro de estos objetivos puede ser trazado tanto en forma estática, durante el desarrollo y el mantenimiento, como dinámica, durante la operación del sistema.

Los objetivos de la misión describen el propósito de alto nivel del sistema a los ojos de los propietarios del sistema y deberían responder a la pregunta de por qué resulta necesario el sistema y qué es lo que el sistema necesita realizar. Los objetivos pueden ser documentados de una manera simple como un conjunto de requerimientos, cuya formalidad es principalmente un aspecto a ser negociado entre el propietario del sistema y el desarrollador. Sin embargo, los objetivos de la misión deben dejar en claro qué constituye exactamente el soporte adecuado de la misión. Decidir respecto de si una organización ha alcanzado su misión no siempre es algo directo [Ellison 99a]:

> *El juzgamiento de la manera en que una misión ha sido satisfecha en forma exitosa o no generalmente se lleva a cabo en el contexto de condiciones externas que puede afectar el logra de la misión. Por ejemplo, supongamos que un sistema financiero sale de servicio durante 12 horas debido a la falta del servicio de energía eléctrica causado por un*

> *huracán. Si el sistema preserva la integridad y confidencialidad de los datos y retoma sus servicios esenciales luego que se ha superado la emergencia ambiental, se puede razonablemente juzgar que el sistema ha satisfecho su misión. Sin embargo, si el mismo sistema sale de servicio de manera inesperada durante 12 horas bajo condiciones normales (o bajo una emergencia ambiental relativamente menor) y priva a sus usuarios de servicios financieros esenciales, el sistema puede razonablemente ser juzgado como que ha fallado en su misión, aún cuando la integridad y confidencialidad hayan sido preservados.*

Qué es lo que constituye exactamente un tiempo de caída para un sistema financiero necesita estar especificado como parte de los objetivos de la misión. Sin embargo, detallar todas las maneras que las amenazas pueden hacer que el sistema financiero sufra una caída es un tema de posterior análisis y documentación. Por supuesto, esto resulta cierto en caso que las amenazas sean o no causadas intencionalmente, razón por la que las amenazas maliciosas sobre la misión se deben analizar y documentar cuidadosamente como parte del TRIAD.

4.3.2.2 Amenazas de la misión

El ambiente operacional del sistema también es el ambiente operacional del atacante, quien puede explotar las vulnerabilidades existentes en los sistemas, en la administración del sistema, y en sus operaciones. La estrategia de supervivencia documenta las características de los ataques que influyen en el diseño. Estas características pueden incluir:

- Perfiles de los atacantes de más alto riesgo
- Vulnerabilidades del ambiente operacional
- Probables blancos y estrategias le permiten al atacante alcanzar sus objetivos
- Escenarios de intrusión detallados en términos de la arquitectura implementada del sistema y de las operaciones soportadas.
- Escenarios de intrusión que apunten a las vulnerabilidades asociadas con el sistema implementado, incluida la infraestructura computacional, la administración del sistema, y las operaciones soportadas.

4.3.2.3 Requerimientos de la supervivencia

La misión y el ambiente operativo generan los requerimientos de supervivencia en términos del comportamiento esperado del sistema y establece restricciones a las respuestas ante ataques. Por ejemplo, la disponibilidad de recursos administrativos del sistema puede estar limitada en algunos lugares. Los *workflows* que comprenden a varias organizaciones podrían requerir de la interoperabilidad entre diferentes arquitecturas de seguridad en lugar de utilizar una infraestruc-

tura común. La estrategia de supervivencia para un misión militar con un requerimiento de bajo tiempo de respuesta podría requerir de la implementación de acciones operacionales alternativas que no dependan de los sistemas impactados. La documentación de los requerimientos de supervivencia puede incluir:

- La respuesta operacional y del sistema esperada para los escenarios de amenazas identificados.
- El impacto operacional de los ataques.
- El tipo de respuesta deseado: recuperación fuera-de-línea, servicio reducido.
- Restricciones respecto a:
    - La asignación de responsabilidades para la implementación de la estrategia a lo largo de múltiples sistemas y organizaciones.
    - La asignación de la estrategia en términos de personas, sistemas, tecnología y operaciones.
    - La asignación de la estrategia en términos del peso dado a la resistencia, reconocimiento, recuperación y adaptación.

4.3.2.4 Arquitectura conceptual

La arquitectura conceptual debe implementar los requerimientos de supervivencia de tal forma que el cliente pueda evaluar y aceptarlos como una manera de atender a sus necesidades. La documentación de la arquitectura conceptual puede incluir:

- *Vistas arquitectónicas*. Pueden documentar la asignación de la respuesta ante amenaza mediante recursos físicos, o la ejecución del flujo de respuesta. Las vistas arquitectónicas tradicionales incluyen la vista del componente y su conector, las cuales se concentran en el comportamiento bajo ejecución del sistema, y la vista de los recursos arquitectónicos, la cual asocia los componentes y conectores con el *hardware* [Clements 02].
- *Suposiciones generales de diseño*. Las suposiciones de diseño pueden ser documentadas como requerimientos de supervivencia, incluyendo las responsabilidades de gestión de fallo correspondientes a la infraestructura, las aplicaciones o las operaciones.
- *Balances arquitectónicos*. Los balances pueden abarcar propiedades funcionales o no-funcionales del sistema tales como desempeño o mantenibilidad.

Otros documentos asociados pueden incluir planes de respuesta frente a desastre o ataque, entrenamiento del administrador de sistema y personal de operaciones, acuerdos en la definición de responsabilidades y respuestas compartidas entre múltiples organizaciones, y acuerdos de calidad de servicio.

4.4 DESARROLLO DE LA ESTRATEGIA DE SUPERVIVENCIA

El principal objetivo de las iteraciones iniciales de TRIAL es la formulación de una estrategia de supervivencia coherente, justificable y asequible. Las iteraciones subsiguientes utilizan esta estrategia como guía para el diseño y mantenimiento del sistema. TRIAD ayuda a los responsables de la toma de decisiones a identificar las opciones arquitectónicas, dimensionar su efectividad, y analizar su impacto sobre las operaciones y las propiedades del sistema, tales como el desempeño. Los análisis realizados como parte de TRIAD sustentan las decisiones incrementales de diseño, como también las justificaciones necesarias para la adquisición, la diligencia debida y la certificación.

La Figura 24 refina el proceso de desarrollo de una estrategia de supervivencia que inicialmente fuera especificado en la Figura 18. *La identificación de amenaza* se necesita para evaluar la robustez de la arquitectura conceptual cuando se utilizan dinámicas de amenaza. La *mitigación de riesgo* se necesita para traducir el análisis de las dinámicas de amenaza en mejoras efectivas y estratégicas sobre la arquitectura conceptual. Esta actividad requiere valorar la vulnerabilidad de la arquitectura conceptual en lo que hace a la falla de la misión y puede exigir la negociación entre diferentes atributos de calidad el sistema.

Por ejemplo, la actividad de identificación de amenaza puede indicar una alta probabilidad de ataques de denegación de servicios basados en red provenientes del exterior. El análisis de dinámicas de amenaza podría indicar la vulnerabilidad arquitectónica frente a tales ataques y la negociación con otros objetivos de la misión tales como un alto rendimiento y usabilidad de los servicios Web. Las actividades de mitigación del riesgo deberían proponer el modo de abordaje para atenuar el impacto de tales ataques, como por ejemplo, el filtrado de tráfico de red, el rastreo de intrusión, o incrementar la capacidad del servidor o de la red.

Un subproducto de este proceso de refinamiento de la estrategia de supervivencia da por resultado una justificación del diseño de la supervivencia del sistema, que incluye la aceptación o rechazo de las alternativas de diseño consideradas. Esta justificación abarca el traceo desde los objetivos de la misión hasta la arquitectura conceptual que ayuda a alcanzar dichos objetivos a pesar de un ataque activo.

Los patrones de ataque de alto nivel permiten la identificación de posibles ataques dentro del contexto de las operaciones del sistema. La aplicación de tácticas de supervivencia sugiere respuestas a estos ataques. Desde ya que la arquitectura conceptual debe resultar apropiada, en última instancia, para las restricciones operacionales del dominio de aplicación.

A continuación se describen las actividades del proceso de refinamiento de la estrategia de supervivencia: *identificación de amenaza*, *análisis de las dinámicas de amenaza*, *mitigación de*

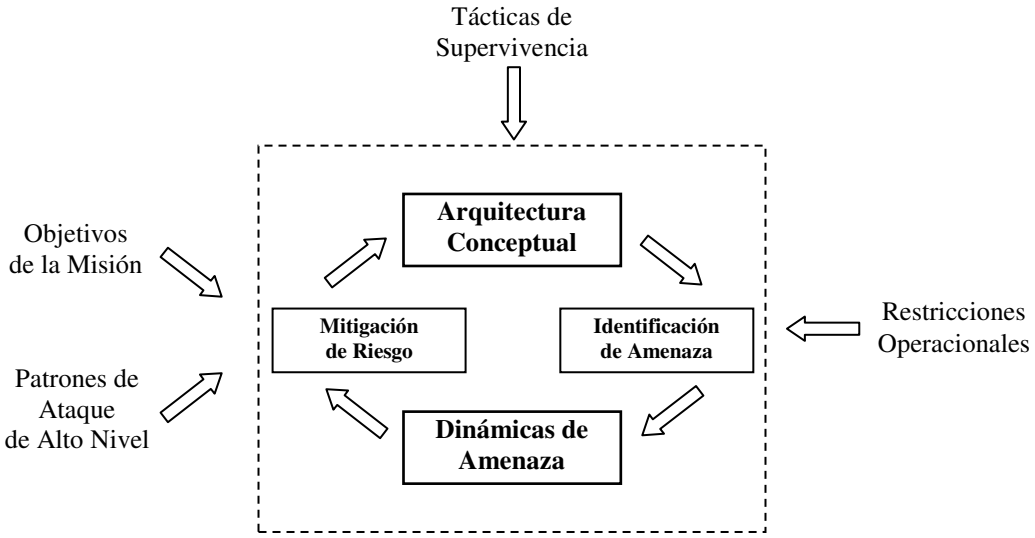
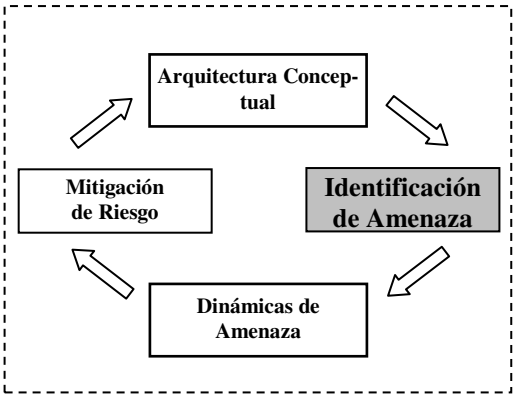


Figura 24. Proceso de refinamiento del proceso de supervivencia.

riesgo, y *refinamiento de la arquitectura conceptual*. Se comienza con la identificación de amenaza, dado que ésta deberá ser la primera actividad durante el desarrollo de una estrategia de supervivencia para un sistema luego que han sido caracterizados los objetivos de la misión. El refinamiento de la arquitectura conceptual se caracteriza al final, debido a que el mismo evoluciona como resultado de las otras tres actividades. A medida que se presentan las actividades del proceso, se describe de qué manera se documenta la estrategia de supervivencia en términos de los objetivos de la misión, las amenazas de la misión, los requerimientos de supervivencia, y la arquitectura conceptual. Esto debería resultar de ayuda tanto para comprender el contexto de una estrategia de supervivencia como para el desarrollo de una estrategia de supervivencia.

4.4.1 Identificación de Amenaza

La creciente confianza depositada por las organizaciones gubernamentales y empresariales en los sistemas de gran escala altamente interconectados amplía las consecuencias de los ataques maliciosos y los compromisos derivados. Además, la complejidad y apertura de estos sistemas al pú-



blico en general incrementa su exposición y vulnerabilidad a la actividad maliciosa. El resultado es que ataques más y más sofisticados están explotando las vulnerabilidades expuestas a una velocidad alarmante. Como se ha visto en los últimos gusanos y virus aparecidos en Internet (por ejemplo, *Melissa*, *Love Letter*, *Code Red*, *Nimda*), los atacantes comparten herramientas y conocimiento para ampliar sus capacidades [CERT 02]. Cada método de ataque aprovecha el conocimiento, la experiencia y el código del método de ataque previo, lo que hace que, irónicamente, el ataque (virus, gusano, etc.) posea más capacidad de supervivencia. Las cada vez más sofisticadas herramientas que disponen los atacantes permiten que individuos relativamente sin experiencia ejecuten ataques muy avanzados.

A esto se agrega que se han visto que los ataques escalan con la intensidad de los conflictos políticos, tales como la guerra de Kosovo, las tensiones entre EEUU y China, y el conflicto entre India y Pakistán [Vatis 01]. En tanto estos ataques generalmente se presentan en forma de desfiguraciones bochornosas de sitios Web, los atacantes están comenzando solapadamente a apuntar hacia la percepción de los usuarios, como por ejemplo los intentos de modificar el contenido de importantes publicaciones de noticias o novedades [Cybenko 02]. Con la actual guerra anti-terrorista, sólo cabe esperarse más ataques de esta naturaleza orientados a socavar la supervivencia de la misión militar. Un reciente informe que analiza las posibilidades del ciberterrorismo concluye que "un ataque semántico sobre un sitio de noticias o de una agencia gubernamental, que hiciera que sus servidores Web proveyesen información falsa en un coyuntura crítica de la guerra anti-terrorista, podría tener un alto impacto sobre la población de los EEUU" [Vatis 01]. En resumen, los ataques realizados por individuos más sofisticados que el *hacker* promedio (por ejemplo, espías industriales o ciber-terroristas internacionales) son más probables y más difíciles de contrarrestar.

Una vista amplia, aunque poco común, de las amenazas incluye el potencial daño resultante provocado por ataques maliciosos, errores del usuario, fallas tecnológicas, y desastres naturales. *El tradicional análisis de fiabilidad a menudo trata con una lista estática de fallas con tasas de fracaso conocidas. El análisis en ese contexto puede conducir a una valoración precisa de la relación costo-beneficio de las estrategias preventivas tales como la replicación del almacenamiento. En cambio, la supervivencia tiene que gestionar una lista no-estática de fallas generadas de manera maliciosa y, a menudo, muy raras*. Los esfuerzos actuales están orientados a limitar el alcance de este análisis a los ataques maliciosos, debido a que las amenazas debidas a actos, fallas o accidentes no intencionales son eventos aleatorios que pueden ser analizado con las técnicas de fiabilidad y tolerancia a fallos existentes. Sin embargo, los ataques maliciosos a menudo incluyen el peor conjunto posible de acciones elucubradas desencadenadas en el momento más inoportuno, que provocan la falla de la misión. A esto se suma que el ambiente de

amenazas es extremadamente dinámico; es muy probable que los ataques que ocurran en los próximos dos años empleen herramientas completamente nuevas para explotar vulnerabilidades existentes no descubiertas.

4.4.1.1 Caracterización del Atacante

La identificación de las amenazas que son relevantes a las operaciones de la organización comprende la caracterización de los tipos de atacantes que muy probablemente amenacen la misión de la organización y los tipos de ataques que estos atacantes muy probablemente realicen. Los atacantes se pueden caracterizar de manera muy general de acuerdo a un conjunto de atributos:

- *Recursos*. Los recursos que se incluyen son económicos, de personal y los niveles de experticia de este personal.
- *Momento*. Un atacante puede tener objetivos de muy corto plazo o puede ser muy paciente y esperar la oportunidad.
- *Herramientas*. El atacante sofisticado puede adaptar herramientas de ataque para cambiar sus rasgos identificatorios y de esta manera evitar su detección, o puede desarrollar herramientas o enviar vía correo electrónico un virus que apunte a un sistema específico.
- *Riesgo*. Un atacante puede buscar publicidad, o un atacante que opera desde el exterior puede no ser intimidado por acciones legales.
- *Acceso*. El acceso de intruso se puede describir en términos de
    - Los mecanismos de acceso utilizados durante el ataque, tal como un *modem dial-up*, una línea tipo DSL, o la Internet.
    - El origen del ataque, tal como desde la interfase externa (*outside*) de un *firewall* conectado a una LAN, o conectado desde un sitio de confianza (*trusted*)
    - La posición organizacional del atacante, si la tuviera, tal como un empleado, un administrador de sistema o un contratado.
- *Objetivos*. Los objetivos del atacante pueden incluir motivaciones políticas, financieras, criminales, militares o personales.

La caracterización de tipos específicos de atacantes escapa al alcance de este trabajo. Existe una plétora de libros que describen los atributos y las técnicas de individuos medianamente sofisticados, pero maliciosos, generalmente llamados *hackers* o *crackers*. La caracterización de atacantes más sofisticados, tales como espías industriales y ciber-terroristas internacionales, por lo general es información sensible y, algunas veces, clasificada [OPSEC 00].

4.4.1.2 Caracterización del Ataque

Los ataques particulares se pueden clasificar de una manera amplia de acuerdo a que estén

basados en personas, en tecnología o en contexto. Estas clases de ataques apuntan, respectivamente a:

- *Deseos, necesidades, capacidades o percepciones de personas*. Ejemplos de esta clase incluyen ingeniería social, ataques semánticos, extorsión y daño físico. Tales ataques pueden explotar la codicia, el miedo o la ingenuidad, la corrupción moral, o personal esencial no-capacitado.
- *Tecnología computacional o de red*. Ejemplos de esta clase incluyen
    - *Ataques basados en red*: ataques sobre la infraestructura de comunicaciones y los servicios soportados; por ejemplo, ataques de denegación de servicio basado en red, incluido denegación de servicio distribuido.
    - *Ataques basados en aplicaciones*: ataques sobre las aplicaciones componentes de la arquitectura tales como un servidor Web, los servicios de correo electrónico, o la infraestructura de soporte de las aplicaciones; por ejemplo, explotaciones que apuntan a vulnerabilidades de un servidor Web, tales como una vulnerabilidad de *buffer- overflow*, para obtener acceso de mayor nivel.
    - *Ataques centrados en los datos*: ataques sobre el flujo de datos o el contenido presentado a través de transacciones. Este tipo de patrones de ataque pueden explotar o corromper datos y servicios o interrumpir o denegar servicios esenciales; por ejemplo, están incluidos ataques que apuntan a relaciones de confianza entre diferentes equipos, o que apuntan a usuarios ingenuos (tal como archivos adjuntos a mensajes de correo electrónico que contienen código malicioso).
- *El contexto en el cual las personas realizan su trabajo*. Ejemplos de esta clase incluyen ataques sobre el soporte del trabajo, la demanda de clientes, el valor de las existencias de la organización, o restricciones legales bajo las que trabajan las personas o las organizaciones. Estos ataques pueden explotar o denegar recursos críticos o dañar el mercado, la capacidad o los activos de la organización.

Los escenarios de intrusión tienen que ver con las interacciones desde el punto de vista del adversario, una vista negativa con respecto a la funcionalidad del sistema, más que a una vista normal de un usuario legítimo, una vista positiva. Se define un *escenario de intrusión como una descripción de la interacción de personas con los sistemas de una manera maliciosa, y por consiguiente, que causa daño a una organización*.

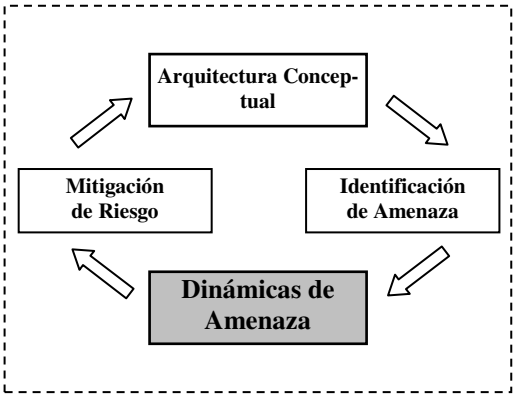
Un escenario de intrusión se puede representar como una secuencia de ataques que conducen a un compromiso específico de la misión del sistema. Un ataque puede o no ser completamente exitoso, pero siempre cambia el estado del sistema de alguna manera. Por otro lado, una intrusión, siempre conduce a un compromiso específico de la misión a través de la ejecución de la

secuencia de al menos ataques parcialmente exitosos. Las intrusiones relacionadas pueden estar convenientemente organizadas en la forma de árboles de ataques en los que la raíz del árbol describe el compromiso de la misión al que contribuyen las intrusiones [Moore 01a].

Sin embargo, los árboles de ataque son de uso limitado en la formulación de una metodología global de supervivencia, dado que al menos ya debe existir una arquitectura de alto nivel para desarrollar un árbol de ataque. Además, cualquier cambio que se haga en la arquitectura como resultado del análisis del árbol de ataque conduce, al menos, a una mejora incremental de la arquitectura. Tales cambios ayudan en el caso que la arquitectura es de suficiente alta calidad, pero sirven de poco en arquitecturas fuera de control. No obstante, los árboles de ataque, y los escenarios de intrusión que ellos generan, proveen una metodología incremental para la formulación de un diseño de bajo nivel y, a fin de cuentas, una implementación que resulta robusta contra ataques probables.

Una técnica relacionada con el uso que hace TRIAD de los escenarios de intrusión, llamada ***casos de abuso*** *(misuse cases* o *abuse cases*) *emplea el concepto de caso de uso del Unified Modeling Language*™ (UML) *para información de seguridad* [McDermott 99, Sindre 00]. La visión más común es que un caso de uso es una especificación general de un conjunto relacionado de escenarios de utilización concretos. Los casos de abuso son a los casos de uso como los escenarios de intrusión son a los escenarios de utilización, es decir, ellos tienen la visión del adversario en lugar de la del usuario. En consecuencia, los casos de abuso pueden ser comprendidos como una manera estándar de describir un conjunto relacionado de escenarios de intrusión. UML identifica explícitamente actores dentro de un diagrama de caso de abuso en correspondencia con los actores dentro de un diagrama de caso de uso. Los casos de abuso describen detalladamente a estos actores maliciosos de acuerdo a sus recursos, experticia y objetivos.

4.4.2 Análisis de Dinámicas de Amenaza



™ Unified Modeling Language es una Marca Registrada de Rational Software Corporation.

Si bien existen variaciones significativas en los detalles de los ataques específicos, son los aspectos comunes de los ataques los que proveen un entendimiento más profundo en la dirección del desarrollo del sistema con supervivencia. Por ejemplo, muchos ataques comparten requerimientos para identificar cuentas de usuario o para bosquejar la topología de la red que soporta el flujo de trabajo. Los ataques se pueden categorizar en términos de la clase de acceso y los privilegios requeridos para ejecutar el ataque: los privilegios de usuario generalmente se requieren para acceder a una aplicación o a datos protegidos; los privilegios de sistema generalmente son requeridos para comprometer *logs* y así afectar las tareas de los forenses; el acceso a la red se requiere para realizar exploraciones y así identificar los servicios disponibles y vulnerables.

Además, en tanto que las vulnerabilidades a menudo son pensadas en términos de las debilidades de los componentes de bajo nivel, las vulnerabilidades a un nivel arquitectónico pueden resultar una amenaza muy superior para la misión de la organización. En general, las vulnerabilidades pueden resultar claras en las operaciones humanas, la arquitectura de la tecnología, o los componentes técnicos particulares. La Tabla 10 proporciona varios ejemplos de qué manera los ataques pueden explotar vulnerabilidades en bruto en un nivel arquitectónico.

Vulnerabilidad	**Impacto**
Administración de sistema distribuido en términos de sitios o en términos de aplicaciones, servidores y redes	Detección y recuperación resultan difíciles de coordinar. Un atacante puede explotar la confusión o las áreas de responsabilidades pobremente definidas.
Múltiples aplicaciones sobre una LAN, cada una con una comunidad externa de usuarios	Un atacante puede exitosamente ganar acceso vía la explotación de una aplicación y luego hacer uso de confianzas compartidas entre las aplicaciones para atacar otros servicios.
Infraestructura compartida	La infraestructura compartida puede impactar en múltiples aplicaciones y sitios
El flujo de trabajo puede atravesar múltiples dominios administrativos	Los errores locales de administración pueden ser explotados. La actividad local del atacante puede no ser observable al sistema objetivo.

Tabla 10. Incremento de la amenaza debido a la vulnerabilidad arquitectónica.

El principal objetivo de las dinámicas de amenaza es el desarrollo de métodos y su verificación destinados a determinar efectivas respuestas estratégicas a las amenazas reales que pesan sobre los sistemas de información a gran escala basados en redes. Las dinámicas de amenaza les permiten a los encargados de la toma de decisiones evaluar el impacto de un potencial ambiente de amenaza sobre el sistema y su misión global. El modelado de las dinámicas de amenaza provee una vista holística de las influencias generales que el ambiente de amenaza puede tener sobre la capacidad del sistema de satisfacer su misión. Esta vista a gran escala permite analizar dinámi-

camente los efectos de los cambios en la actividad del atacante, las respuestas del ambiente operacional frente a la actividad del atacante, los cambios producidos en las operaciones o la arquitectura del sistema, o la disponibilidad de nuevos datos que caractericen las amenazas percibidas desde una nueva óptica. El análisis de las dinámicas de amenaza clarifica el rol que la tecnología posee en la consecución de la misión de la organización en el sentido más amplio.

4.4.2.1 Trasfondo de las Dinámicas de Sistema

Las dinámicas de sistema fueron desarrolladas por Jay Forrester para demostrar de qué manera se debía utilizar un modelo de la estructura de un sistema de actividad humana y de las políticas utilizadas para controlarlo, para lograr una comprensión más profunda de la operación y comportamiento de dicho sistema [Forrester 61]. Las dinámicas de sistema se han venido utilizando en forma extendida como una herramienta general de modelado que permite una mejor comprensión de la estructura y dinámica de los complejos sistemas basados en seres humanos, particularmente dentro del área de estrategia de comercialización y políticas públicas [Sterman 00, Wolstenhome 90].

Las dinámicas de sistema se pueden definir como un método para modelar y analizar el comportamiento holístico de sistemas administrados y complejos a medida que evolucionan a lo largo del tiempo. Los sistemas administrados incluyen cualquier sistema que controla personas, o que trata de controlarlas de alguna manera. *El objetivo de las dinámicas de sistema es la de comprender de qué manera la información de retroalimentación gobierna el comportamiento del sistema, y diseñar estructuras de retroalimentación y políticas de control que mejoren la administración y la operación del sistema*. Coyle define las dinámicas de sistema en términos de la ingeniería de control como "la aplicación de la actitud de un ingeniero de control en el mejoramiento del comportamiento de la dinámica dentro de los sistemas administrados" [Coyle 96]. En tanto que los ingenieros de control diseñan sistemas mecánicos tales como sistemas de calefacción central o de pilotos automáticos para la aviación, los ingenieros de dinámicas de sistema diseñan políticas de control para sistemas basados en seres humanos tales como el sistema de la justicia criminal o la seguridad nacional. Es más, las dinámicas de sistema se fundamentan en las teorías de dinámicas no-lineales y el control de retroalimentación conocidos desde ya hace muchos años por matemáticos, físicos e ingenieros.

Las dinámicas de sistema emplean el término sistema en forma amplia para incluir cualquier colección de elementos interactuantes que se organizan con un propósito dado. Son particularmente útiles para el modelado y el análisis de sistemas con un alto grado de dinámica compleja. La complejidad estática (o combinatoria) surge cuando se intenta realizar una selección óptima entre un número abrumador de posibilidades, como se ha visto que sucede cuando se planifican

los vuelos y las tripulaciones en una gran compañía aérea. En contraste, la complejidad dinámica surge a partir de la naturaleza de las interacciones entre los elementos del sistema a lo largo del tiempo, especialmente la velocidad e intensidad de estas interacciones. La información de retroalimentación, los retardos, la no-linealidad, la incertidumbre y la volatilidad de las respuestas de comportamiento estimulan la complejidad de una comprensión acerca de cómo se comportan los sistemas dinámicos, especialmente en el largo plazo.

La forma más simple de describir el problema cualitativamente y de analizar las dinámicas de sistema es el ***diagrama de influencia***. La Figura 25 muestra dos diagramas de influencia muy simples, uno que representa un sistema de calefacción central y el otro que representa el efecto inherente de la tasa de nacimientos en el crecimiento de la población. Las variables del diagrama representan los elementos del sistema involucrados. Los elementos del sistema pueden ser animados o inanimados, tangibles o intangibles. Los elementos que se muestran en itálicas son, a los propósitos del análisis que sigue, factores (o parámetros) constantes que actúan como entradas (*inputs*) en el cálculo de las variables del sistema. Las flechas etiquetadas con signos representan las interacciones del sistema, donde el signo indica la influencia de la variable en el origen de la flecha sobre la variable hacia la cual apunta la flecha.

- Una influencia positiva (+) indica que si el valor de la variable de origen se incrementa, entonces el valor de la variable a la que se apunta se incrementa por encima de lo que de otra manera hubiera estado, manteniéndose todas las demás cosas iguales. Y, si el valor de la variable de origen se decrementa, entonces el valor de la variable a la que se apunta se decrementa por debajo de lo que de otra manera hubiera estado, manteniéndose todas las demás cosas iguales. Por lo tanto, en el primero de los diagrama de influencia de la Figura 25, para una configuración particular del termostato, a medida que la tasa de entrada de calor se incrementa (decrementa), entonces la temperatura de la habitación se incrementa (decrementa) por encima (debajo) de lo que hubiera estado.
- Una influencia negativa (-) indica que si el valor de la variable de origen se incrementa, entonces el valor de la variable a la que se apunta se decrementa por debajo de lo que de otra manera hubiera estado, manteniéndose todas las demás cosas iguales. Y, si el valor de la variable de origen se decrementa, entonces el valor de la variable a la que se apunta se incrementa por encima de lo que de otra manera hubiera estado, manteniéndose todas las demás cosas iguales. Por lo tanto, a medida que la temperatura de la habitación se incrementa (decrementa), entonces la tasa de entrada de calor se decrementa (incrementa) por debajo (encima) de lo que hubiera esperado el sistema de calefacción central.

Dos conductores claves del comportamiento de la dinámica son los ciclos de retroalimentación y los retardos de tiempo.

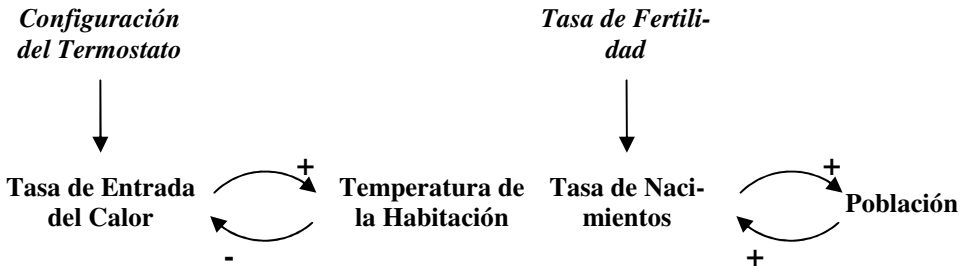


Figura 25. Diagramas de Influencia simples.

Los *ciclos de retroalimentación* se pueden auto-reforzar (+) o se pueden auto-limitar (-). La polaridad de un ciclo de retroalimentación está determinada por la "multiplicación" de los signos a lo largo del camino del ciclo. El sistema central de calefacción de la Figura 25 es auto-limitante dado que el mismo posee un número impar de signos negativos a lo largo de su camino.

Los ciclos auto-limitantes describen aspectos de un sistema que tiende a conducir los valores variables a algún estado objetivo. En el caso del sistema de calefacción central, el estado objetivo es una temperatura de la habitación igual al valor establecido en el termostato. En general, los ciclos auto-limitantes describen aspectos que se oponen al cambio, y que usualmente implican la auto-regulación mediante la adaptación a influencias externas. Claro está que estos aspectos pueden ser o no deseables; por ejemplo, recientes estudios muestran que la reducción de contenido de nicotina en los cigarrillos, supuestamente para beneficiar la salud del fumador, sólo da por resultado que las personas fumen un número mayor de cigarrillos y que realicen largas y profundas pitadas para satisfacer sus necesidades de nicotina; en cambio, un ejemplo de un ciclo auto-limitante beneficioso es el uso de una defensa activa de la red, la cual reconoce y se recupera luego de ataques maliciosos sobre la red para mantener un nivel deseado de seguridad o supervivencia.

Los ciclos que se auto-refuerzan describen aspectos del sistema que tienden a conducir las variables a un aumento sostenido o a una reducción sostenida. El segundo diagrama de influencia de la Figura 25 muestra un ciclo que se auto-refuerza debido al número par de signos negativos a lo largo de su camino (en este caso cero se considera par). Los ciclos que se auto-refuerzan pueden ayudar a explicar el crecimiento explosivo o el colapso implosivo de un sistema; por ejemplo, la carrera en el armamento nuclear fue un ejemplo de retroalimentación auto-reforzado por el cual la URSS construía armas nucleares para contrarrestar la amenaza nuclear impuesta por los EEUU; esto incitaba al permanente incremento del arsenal de armas nucleares de los EEUU, seguido por un incremento aún mayor por parte de los soviéticos, dando por resultado la explosiva fabricación de armas nucleares.

Microsoft e Intel se beneficiaron con el explosivo aumento derivado de haber sido protagonistas en el inicio del ascenso de las computadoras personales, lo cual motivó que los fabricantes de

software apuntaran a la plataforma Windows/Intel, lo que trajo aparejado el rol dominante de estas compañías.

Los ciclos auto-reforzantes también pueden ayudar a explicar el ascenso y la caída de los gusanos de computadora con capacidad de auto-replicación, tales como los gusanos *Code Red* y *Nimda*, que causaran muchos problemas durante el verano del año 2001, pero en escalas de tiempo mucho más comprimidas que en los ejemplos anteriores.

Los *retardos de tiempo* pueden hacer que el comportamiento de los sistemas parezca errático. Estos retardos pueden separar la causa del efecto de tal manera que haga que el efecto a largo plazo resulte muy diferente del efecto a corto plazo. La consideración de los retardos de tiempo dentro de los ciclos de retroalimentación, o la interacción de múltiples ciclos de retroalimentación, ayuda a explicar lo que pareciera un comportamiento contrario al esperado intuitivamente.

Los retardos de tiempo en los ciclos auto-limitantes pueden crear inestabilidad y oscilación tal como el que se observa en el pare y arranque del tráfico o al lograr que una ducha entregue agua a la temperatura apropiada.

Un ciclo de retroalimentación puede amplificar o moderar la influencia de otro ciclo de retroalimentación. Cuando se toma una vista estrecha del sistema, el analista solamente ve parte de toda la imagen, resultando perfectamente explicable que el comportamiento parezca errático e impredecible.

4.4.2.2 Dinámica de Amenaza para la Supervivencia

Se han publicado muy pocos trabajos en los que se apliquen las dinámicas de sistema al estudio de la efectividad de la tecnología de la información. Uno de los pocos trabajos que se encuentran disponibles describe un abordaje que utiliza las dinámicas de sistema para el estudiar el impacto que introduce un sistema de información de administración sobre la misión de la organización [Wolstenholme 93]. El autor desarrolla dos casos de estudio para evaluar el impacto operacional de un sistema de logística militar sobre un sistema de control y comando táctico en el campo de batalla. Argumenta que el abordaje "tiene mucho que ofrecer en las fases de diseño de Sistema de Información de Administración, y la difusa (y a menudo interactiva) frontera entre el diseño y valoración. La aptitud de la técnica de incorporar datos subjetivos en estas fases resulta particularmente ventajoso". Este trabajo está relacionado con esfuerzo del equipo de desarrollo del TRIAD, proveyendo algunas evidencias de su valor general y practicabilidad. Sin embargo, se desconoce la existencia de algún trabajo que utilice dinámicas de sistema para estudiar explícitamente el ambiente de amenaza o su impacto sobre las operaciones del sistema.

No obstante, se puede afirmar que las dinámicas de sistema proveen el basamento para el desarrollo de métodos y herramientas que ayuden a los ingenieros a comprender, caracterizar, y

comunicar el impacto de un ambiente de amenazas maliciosas sobre las operaciones organizacionales y del sistema y de sus respectivas misiones. Los sistemas de información interconectados a gran escala están sujetos a volatilidad, no-linealidad, incertidumbre y retardos de tiempo que se agregan a su complejidad dinámica, y en los que se vuelve muy difícil dar por hecho su seguridad o supervivencia.

- *Volatilidad*. El desarrollo cada vez más rápido de herramientas de ataque y la coordinación de la comunidad de atacantes auspicia un ambiente de amenaza muy volátil para los sistemas de información de negocios y militares [CERT 02]. Dar por hecho la seguridad y la supervivencia de tales sistemas demanda de técnicas que se puedan aplicar como parte de un diseño evolutivo y de un ciclo de vida de mantenimiento.
- *No-linealidad*. La misión de la organización puede volverse drásticamente más vulnerable sólo debido a pequeños incrementos en la capacidad del atacante o de pequeños cambios en las políticas, controles o arquitectura del sistema. Esta no-linealidad hace que el mantenimiento de la seguridad y de la supervivencia aún de sistemas distribuidos relativamente simples resulte muy dificultoso debido a la volatilidad del ambiente de amenaza y a la naturaleza de la defensa activa de red.
- *Incertidumbre*. Si bien cada vez resulta más fácil contar con datos exactos referidos a accidentes y vulnerabilidades, todavía existen importantes brechas en nuestra comprensión del comportamiento del intruso, lo que crea un no despreciable grado de incertidumbre. El análisis de dinámicas de amenaza saca provecho de la disponibilidad de tales datos cuando éstos existen, pero no depende de ellos a los fines de proporcionar percepciones útiles en el impacto del ambiente de amenaza sobre las operaciones del sistema. El análisis de las dinámicas de amenaza, y su fundamento en las dinámicas de sistema, se puede realizar de una manera cualitativa, cuantitativa o combinada [Coyle 00].
- *Tiempo de retardo*. A menudo los principales retardos se producen entre el momento en que el atacante inicia una actividad maliciosa y el momento en que comprendemos el alcance completo de esa actividad. Tales retardos tornan muy dificultosa la implementación de contramedidas estratégicas y la valoración de su efectividad, especialmente cuando se hace necesaria la re-configuración en tiempo real a los fines de frustrar al adversario.

En tanto que las dinámicas de sistema son ampliamente aplicables, resultan más ventajosas en aquellos sistemas que utilizan información derivada para ejercer control de retroalimentación sobre sus recursos. Tal control de retroalimentación es una técnica crítica para la construcción de sistemas de información con capacidad de supervivencia. El método de la defensa activa monitorea la actividad de ataques y responde a través de una variedad de técnicas de recuperación y adaptación para asegurar el éxito de la misión. *En consecuencia, los sistemas con capa-*

cidad supervivencia controlan sus recursos de información basándose, en parte, en la retroalimentación tomada de la actividad ataque-monitoreo. Las dinámicas de sistema ayudan a representar y analizar tal control de retroalimentación, pero generalmente no asumen la presencia de agentes hostiles.

La Figura 26 ilustra algunos conceptos de dinámicas de sistema descriptos en el contexto de las amenazas a los sistemas basados en Internet. La figura describe un ciclo de retroalimentación como un aspecto del comportamiento para controlar la vulnerabilidad de los sistemas basados en Internet [Arbaugh 00]. Comenzando en el elemento "Efectividad en la explotación de la vulnerabilidad" en el extremo inferior izquierdo de la figura, vemos que la efectividad influencia de manera positiva la tasa de publicaciones referidas a la vulnerabilidad, en el sentido que un incremento de la efectividad conduce a un incremento en la publicación (tal vez debido al incre-

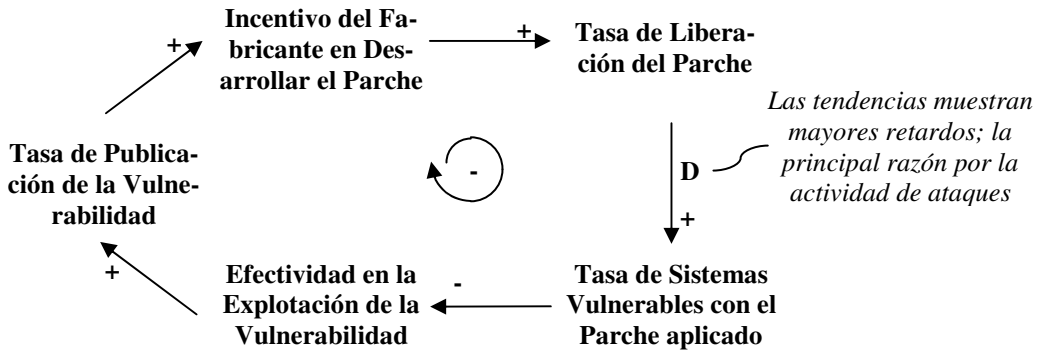


Figura 26. Un ciclo de retro-alimentación para el control de vulnerabilidad.

mento de la publicación conlleva un creciente incentivo en enmendar errores y dejar disponibles los parches relevantes. Estos conducen a la aplicación de parches en los sistemas, lo cual a su vez reduce la efectividad en la explotación de la vulnerabilidad. La demora en la aplicación del parche, indicada con la "D" a lo largo de la flecha del lado derecho de la figura, es una tendencia que ha sido descripta como la principal razón de la fuerte actividad de ataques basados en Internet, y de la vulnerabilidad general de la Internet. No obstante, el ciclo de retroalimentación general descripto es un equilibrio (indicado por el símbolo negativo del ciclo en el centro), en el que la aplicación del parche generalmente ayuda a controlar la vulnerabilidad general de Internet.

Los diagramas de influencia se pueden componer, como se ilustra en la Figura 27. El lado derecho de la figura muestra el diagrama de influencia anterior. El lado izquierdo muestra el ciclo de retroalimentación que describe el efecto de la tasa de publicación de vulnerabilidad respecto de la publicación de las herramientas de explotación y, finalmente, respecto de la explotación de la vulnerabilidad por parte del atacante. Este es un ejemplo de un ciclo de retroalimentación autoreforzante, como lo indica el símbolo positivo del ciclo del centro. Esta figura ilustra un debate

en curso dentro de la comunidad de Internet referido a si publicar datos de vulnerabilidad o entorpecer la seguridad general de la Internet. Análisis recientes indican que los retardos en la aplicación de parches son la principal causa de la vulnerabilidad en Internet, mientras que la publicación de datos de vulnerabilidad es una fuerza de incidencia secundaria [Arbaugh 00].

El diagrama, por supuesto, no ayuda a resolver el debate, dado que es de naturaleza estrictamente cualitativo.

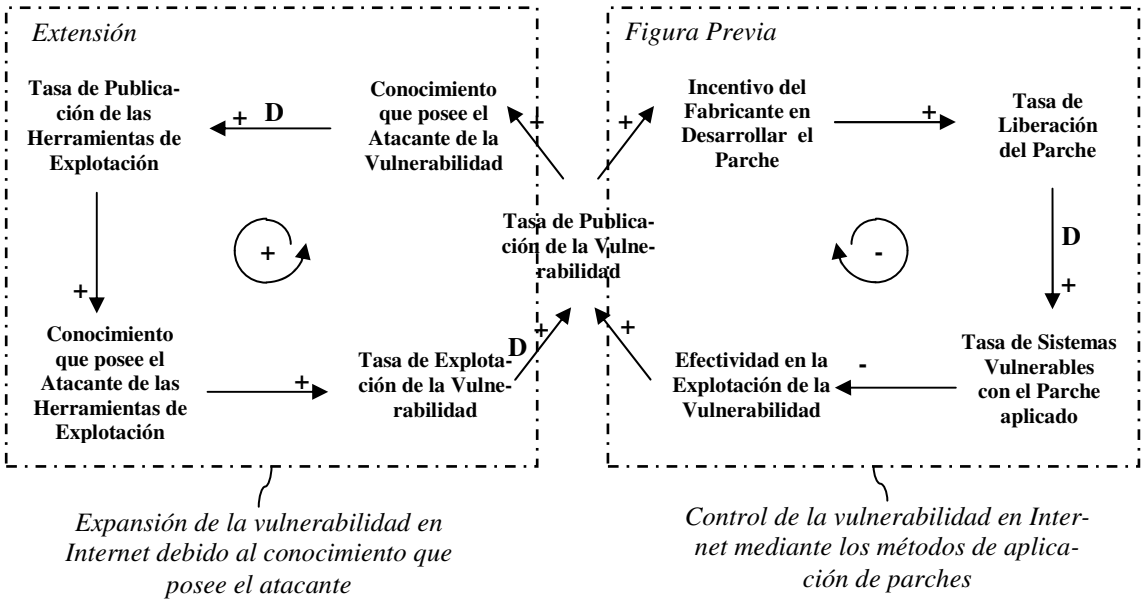
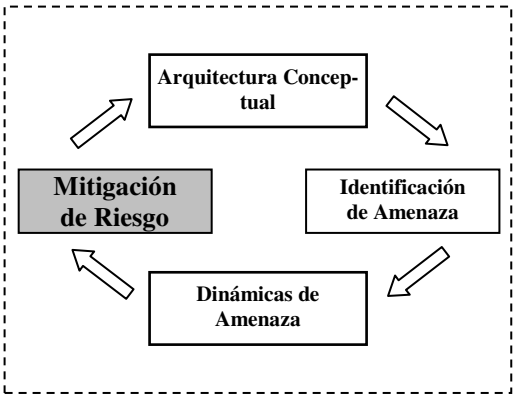


Figura 27. Los efectos de la publicación de vulnerabilidades sobre la vulnerabilidad en la Internet.

4.4.3 Mitigación de Riesgo



Las dinámicas de amenaza ayudan a desagregar la influencia de los ataques y las contramedidas en la capacidad del sistema en alcanzar la misión de la organización. Las influencias indican los puntos débiles de la arquitectura conceptual y el equilibrio que debe lograrse a fin de alcanzar una solución efectiva y realizable. Las soluciones destinadas a reforzar los puntos

débiles y a tomar decisiones en vistas a lograr un equilibrio adecuado pueden ser de naturaleza activa o pasiva.

Las técnicas preventivas tales como autenticación, autorización (control de acceso) y encriptación incrementan la resistencia frente a ataques o son generalmente pasivas. La respuesta de un sistema ante el ataque en términos de recuperación de servicios o continuación de las operaciones (posiblemente degradadas) puede resultar en un cambio de la configuración del sistema o de la política de seguridad, por ejemplo, reforzar la autenticación o estrechar el control de acceso. Las respuestas dentro de este contexto son activas en el sentido que el sistema debe detectar el ataque y luego activamente reconfigurar las operaciones del sistema para asegurar el éxito de la misión en presencia del ataque.

La arquitectura del sistema tiene que soportar la continuidad del servicio en presencia de ataques mediante servicios redundantes o servicios de recuperación de datos de tal manera de restaurar un servicio pleno luego de ocurrido el ataque. La selección y la importancia que se le asigne a los métodos pasivo o activo resultan críticos para la estrategia de supervivencia.

4.4.3.1 Limitaciones de la tecnología actual

Un método reactivo aislado para construir y mantener la seguridad y la supervivencia del sistema está condenado al fracaso debido a las limitaciones prácticas de achicar la ventana de exposición de los sistemas vulnerables [Arbaugh 00, Schneier 00b]. *De hecho, ningún nivel de endurecimiento puede asegurar que no ocurrirán intrusiones en sistemas con límites débilmente definidos. La detección de ataques y una exitosa recuperación son parte esenciales de la supervivencia del sistema.*

Desafortunadamente, la tecnología de detección de intrusión sólo puede resolver una pequeña parte del problema, al menos en sus formas actuales. Esta tecnología apunta a la identificación de ataques basados solamente en computadora o en red. Los ataques que "vuelan por encima del radar" de la tecnología de detección de intrusión, tales como ingeniería social y ataques físicos, necesitan ser tomados tan seriamente como los ataques tecnológicos [Anderson 01]. La tecnología de detección de intrusión no identifica la actividad correlacionada de los ataques multi-etapa –ataques que pueden incluir coerción, corrupción, o desengaño de personas sumados a la explotación de vulnerabilidades tecnológicas. En general, los ataques pueden tener como blanco usuarios y componentes internos del sistema, como así también sistemas externos de confianza y comunidades de usuarios. Hacer caso omiso al factor humano podría ser muy engaño y traer aparejadas importantes brechas en las defensas del sistema.

Otro problema con la tecnología de detección de intrusión actual es la alta tasa de falsos positivos (detectar un ataque cuando no ocurre nada) y falsos negativos (no detectar un ataque que en

realidad está teniendo lugar). Los falsos positivos requieren de un analista humano para revisar los *logs* de auditoría e identificar si una intrusión se ha producido efectivamente. Si bien este analista puede ser una parte necesaria de un diseño de sistema con capacidad de supervivencia, la tecnología de detección de intrusión que posee altas tasas de falsos positivos puede introducir una carga innecesaria sobre los administradores, y realmente puede ser un perjuicio para la supervivencia de la misión en el largo plazo.

Los falsos negativos son tal vez aún más perniciosos que los falsos positivos. Los analistas del CERT/CC han estado viendo cada vez más ataques furtivos que "vuelan por encima del radar" de la tecnología de detección de intrusión. Una simple prueba ejecutada una vez por día puede permitirle a un adversario paciente descubrir la red de una organización de manera tan efectiva como un escaneo en profundidad, y sin ser detectado. A lo que se suma que la mayoría de los patrones de ataque en los que se basa la tecnología de detección de intrusión no representan la actividad correlacionada de un atacante capaz sino simplemente algún punto intermedio de un ataque que a menudo es perpetrado por un novicio relativamente experimentado. Estos patrones no representan de manera completa ni segura el comportamiento de los atacantes sofisticados y motivados y, en consecuencia, no son un basamento adecuado para identificar la amenaza que puede imponer o detectar los ataques que ellos están perpetrando.

La tecnología de intrusión existente fomenta la aplicación en el caso de la supervivencia del mismo método *bottom-up* que se discutiera en al inicio. Sin una visión de la misión en forma global, una organización puede desperdiciar mucho tiempo y recursos intentando detectar y analizar ataques que no tienen impacto en su habilidad de éxito. Sin embargo, la detección de intrusión será probablemente una parte cada vez más importante en la construcción de sistemas con capacidad de supervivencia. Un informe reciente sobre el estado del arte en las tecnologías de detección de intrusión recomienda que, entre otras cosas, las tecnologías futuras deberían integrar una mayor diversidad de datos de ataque para disminuir las imprecisiones, defender contra ataques que son más sofisticados que los producidos por el *hacker* promedio, e integrar el análisis humano como parte del diagnóstico de eventos [Allen 00].

Si bien estas recomendaciones resultan importantes, se sugiere llevarlas un paso más allá para tratar directamente con las limitaciones inherentes de una solución estrictamente tecnológica. *Las organizaciones se deberían concentrar en la detección de intrusión y responder de una manera holística mediante la integración de una detección de intrusión de conjunto y la capacidad de respuesta con políticas y procedimientos de la organización, como así también con la tecnología.*

4.4.3.2 Tácticas de Supervivencia

El empleo de tácticas de supervivencia en TRIAD deriva de la noción de una táctica arquitectó-

nica desarrollada en el Software Engineering Institute [Bachmann 02]. El objetivo de este trabajo es describir de qué manera atributos de calidad tales como el desempeño o la modificabilidad ejercen influencia sobre el diseño arquitectónico, de qué manera se pueden codificar estas influencias, y cómo se pueden utilizar estas nociones para analizar arquitecturas. Los autores definen una táctica arquitectónica como "una decisión de diseño que ayuda a lograr una respuesta calidad-atributo y que es motivada por un modelo de análisis calidad-atributo". En lo que respecta al desempeño y a la latencia, tales decisiones de diseño incluyen el tamaño y número de servidores y la administración de la concurrencia en el servidor en base al análisis basado en modelos de colas y planificación. Las tácticas de modificabilidad incluyen la administración de la información pública y privada correspondiente a un módulo para localizar las modificaciones esperadas. El análisis asociado examina las dependencias entre sistemas, la probabilidad de cambios, y el impacto al realizar estos cambios.

Las tácticas de supervivencia, una clase particular de las tácticas arquitectónicas, codifican las decisiones de diseño que ayudan a mitigar el riesgo asociado con un ambiente de amenazas maliciosas. En tanto existe una variación significativa en los detalles de los ataques específicos, son los atributos comunes de los ataques que proporcionan la mejor comprensión en dirección al desarrollo de un sistema con capacidad de supervivencia. *Las tácticas de supervivencia describen de qué manera las decisiones de diseño mitigan el riesgo asociado con todos los ataques que comparten estos atributos comunes*; por ejemplo, una técnica popular de ataque es explotar una vulnerabilidad sobre un servidor accesible por Internet tal como un servidor FTP o Web, y luego aprovechar el acceso logrado a partir de dicha explotación para atacar sistemas relacionados. Esta técnica corresponde a la siguiente táctica de supervivencia: el empleo de *firewalls* con una configuración de zona desmilitarizada (DMZ) limita el acceso del atacante a otros sistemas luego de la penetración inicial.

Las tácticas de supervivencia también resultan útiles para la especificación de esquemas de recuperación luego de intrusión. El sistema y los *logs* de aplicación deben permitir el análisis que sigue a un ataque para identificar el alcance y el impacto, y generar el plan de recuperación detallado. Las tácticas de supervivencia orientadas a la infraestructura computacional, tales como servicios replicados, mejoran la recuperación pero pueden incrementar los costos de administración del sistema. Para aquellos procesos que comprenden múltiples emplazamientos u organizaciones, una táctica de recuperación puede ser restaurar servicios en forma local y luego sincronizar el conjunto de sistemas. Se debe observar que una táctica de supervivencia puede ayudar a enfrentar ataques nuevos, nunca antes vistos, en caso que estos nuevos ataques compartan los atributos comunes atendidos por esta táctica. La Tabla 11 describe otro ejemplo de tácticas de supervivencia.

Muchas de las anteriores tácticas de supervivencia han sido útiles en el análisis de la capacidad de supervivencia en los sistemas del mundo real, como se las aplicara en el Método *Survivable System Analysis* [Mead 00a]. Una preocupación particular en las aplicaciones precedentes al SSA ha sido *la asignación de responsabilidades en la preservación de la misión a través de múltiples organizaciones*.

Las respuestas a ataques comprenden una combinación de resistencia, reconocimiento y reacción ante eventos. Algunas respuestas dependen de cambios operacionales inmediatos con la organización involucrada, mientras que las demás utilizan una arquitectura o tecnología específica para limitar el impacto del ataque sobre las operaciones. *Decidir respecto de la respuesta óptima depende de una variedad de factores, los que incluyen*:

- *Sensibilidad ante la falla operacional*. ¿Las operaciones esenciales tienen un requerimiento de tiempo de respuesta próximo al tiempo real? ¿Cuál es el impacto del acceso limitado a los datos? Para un *workflow* distribuido, ¿qué procesamiento local puede continuar si la red se ve comprometida?
- *Impacto operacional para las clases de respuestas*. Las opciones de respuestas incluyen continuar las operaciones desencadenando la recuperación en el *background*; reducir el servicio en términos de la comunidad de usuarios o lo funcionalmente soportado; continuar locamente con las operaciones y, eventualmente, sincronizando y sacando el sistema de operación hasta que la recuperación esté completa y la amenaza contenida.
- *Restricciones operacionales*. Las restricciones pueden incluir habilidades del personal en cuanto a operar y administrar el sistema y las limitaciones impuestas por los sistemas legales, acuerdos contractuales, o limitación en la autoridad.

Los ataques a menudo explotan errores en un componente de software del sistema. Un error explotable generalmente se denomina vulnerabilidad del sistema. La ejecución del ataque genera la falla asociada con el error. En consecuencia, muchas de las tácticas de supervivencia comprenden la administración de fallas. Un objetivo común para la administración de fallas es la de ocultarle a las aplicaciones las fallas de red o de hardware. Pero los ataques pueden generar combinaciones raras de fallas independientes o explotar una débil integración entre los componentes. Las tácticas de supervivencia pueden requerir que algunas fallas sean visibles a las aplicaciones; por ejemplo, un ataque combinado sobre una red táctica militar podría requerir que las aplicaciones que utilizan la red ajusten su comportamiento en caso que el ancho de banda disponible se haya visto reducido.

La introducción de cambios correctos y seguros en la configuración del sistema requiere de información de estado del sistema exacta y a tiempo. Los componentes administrativos *self-*

repair tienen que monitorear la red y realizar el sensado de fallas que podrían impactar sobre la distribución e integridad de la información de estado del sistema y, potencialmente, limitar las re-configuraciones cuando existe baja confianza en los datos.

Tipo de ataque	Estrategia del atacante	Tácticas de supervivencia
Denegación de servicio	Orientado a servidor o red. Compromete las operaciones de un servicio de infraestructura tal como un servidor de directorio o la consola de administración de red, lo cual impacta a un amplio rango de servicios computacionales. Los ataques de denegación de servicio no necesariamente requieren del acceso de usuario al sistema.	• Arquitectura de red: filtro de paquetes, rastreo al origen del intruso, capacidad en reserva, servicios distribuidos. • Arquitectura de infraestructura: replicación, recuperación acelerada. • Arquitectura de aplicación: servicios vía proxy para monitoreo de contenido.
Compromiso del contenido de aplicación	Orientado a aplicaciones o servicios de administración de datos. Ejemplos: virus de correo electrónico, uso de ingeniería social para inducir a un empleado a ingresar datos inválidos, un ataque exitoso sobre un sitio de confianza inserta información comprometida dentro del flujo de datos.	• Aplicación: filtrado de virus y escaneo, testeos repetidos sobre la integridad y consistencia de los datos aún cuando los datos provengan de sitios de confianza, monitoreo de acceso de datos y bloqueo de actividad sospechosa. • Personal: entrenamiento.

Tabla 11. Tácticas de supervivencia enfrentando tipos de ataques.

Dónde ubicar dentro de la arquitectura la responsabilidad del reconocimiento del ataque y la respuesta asociada depende de la clase de fallas generadas por la clase de interés del ataque. La administración de red y la detección de intrusión basada en red apuntan tanto a ataques de denegación de servicio como a ataques que explotan el protocolo IP (*Internet Protocol*). Los sistemas de detección de intrusión basados en *host* se concentran en ataques que explotan vulnerabilidades del sistema o del servidor. La detección y respuesta a los ataques que dirigidos al contenido de datos dentro de un intercambio podrían ser responsabilidad de las aplicaciones que comprenden la semántica de la transacción.

La experiencia con ataques reales sobre sistemas a lo largo de los años pone énfasis en la necesidad de considerar la imagen global, que incluya tanto la tecnología como su ambiente operacional, a los fines de desarrollar soluciones fuertes y efectivas en su costo / beneficio [Anderson 01, Schneier 00a]. *Las técnicas que se indican a continuación resultan útiles durante el desarrollo de las tácticas de supervivencia, ya sean individualmente o combinadas. Estas técnicas pueden ser implementadas manualmente (a través de procedimientos realizados por humanos), automáticamente (a través de la tecnología), o una combinación de ambos*.

- *Redundancia*. Anderson define la redundancia como "mantener un margen de componentes disponibles o información duplicada para reemplazar los activos dañados o com-

prometidos" [Anderson 99]. La replicación de componentes, conexiones y/o datos, a menudo no en la misma localización de la copia original, combinada con una buena administración de la replicación puede permitir la continuación del servicio cuando falla o se ve comprometida la copia original.

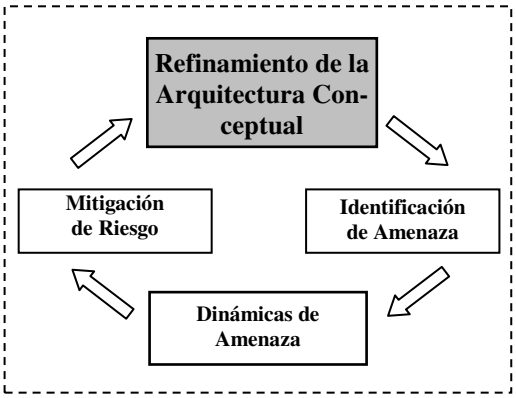
- *Diversidad*. La diversidad comprende el uso de diferentes métodos, componentes y/o plataformas para evitar que los atacantes exploten las mismas vulnerabilidades repetidamente; como ejemplo se incluyen el empleo de diferente hardware, diferentes sistemas operativos, e incluso diferentes técnicas de programación tal como programación de n-versiones. Cuando se emplea esta diversidad en diferentes puntos de entrada al sistema se puede incrementar el factor trabajo del atacante.
- *Señuelo*. El fiasco puede ser utilizado por el defensor al igual que una amenaza a la supervivencia empleada por un adversario. Anderson define al fiasco, en referencia al aseguramiento de la supervivencia, como un "artificio que aspira a inducir comportamientos en el enemigo que pueden ser explotadas" [Anderson 99]. El ejemplo más común es el uso de un conjunto de información equívoca para hacerle gastar tiempo al atacante en tanto que otros mecanismos montan una respuesta apropiada al ataque. Esta información equívoca a menudo se conoce eufemísticamente como *honeypot*.
- *Identificación / Autenticación*. NSA define la autenticación como la verificación de un identidad reclamada como legítima y perteneciente a quien la reclama [NCSC 91]. La mayoría de los tipos de control de acceso requieren de una identificación y autenticación de usuarios seguras. La técnica más común por muchos utilizada es nombre de usuario / contraseña; también son posibles técnicas más robustas utilizan biométricas, *tokens* y firmas criptográficas.
- *Detección de intrusión*. Las intrusiones requieren "tanto de un acto evidente del un atacante como de una manifestación observable por parte de la víctima, que resulte de ese acto" [McHugh 00]. El objetivo de la detección de intrusión es el de observar y reportar acerca de la manifestación de la intrusión de un atacante. El reporte puede realizarse manualmente, mediante el análisis humano, o automáticamente, utilizando sistemas de detección de intrusión. La detección de intrusión puede tener lugar en tiempo real o a través del análisis *off-line* de los datos de auditoría de actividad del sistema registrados en forma separada. Las intrusiones se pueden detectar mediante la búsqueda de firmas de ataques conocidos (chequeo de virus es un ejemplo común) o la búsqueda de anomalías –actividad del sistema que no se adecua a patrones de utilización "normales". Los sistemas de detección de intrusión disponibles en la actualidad apuntan tanto a tráfico de red de bajo nivel o a la utilización de la aplicación a alto nivel [McHugh 00]. Los ejecu-

tables que chequean la integridad del sistema en base a parámetros del sistema también son una forma de detección de intrusión.

- *Recuperación / Adaptación*. La recuperación y la adaptación son las respuestas del sistema a la detección de intrusión. La recuperación generalmente es el término que se aplica a la reparación o reemplazo de datos, componentes, o comunicaciones dañadas debido a la intrusión. La adaptación generalmente comprende la planificación y reconfiguración a largo plazo destinadas a evitar intrusiones similares a futuro. Los ejemplos varían desde complejas técnicas tales como asignación dinámica de recursos hasta activos y actividades de alta prioridad y la auto-organización de agentes autónomos distribuidos [Anderson 99] hasta técnicas simples como restauración a partir de respaldos históricos y corrección de errores.
- *Separación física, lógica, criptográfica y temporal*. La seguridad a nivel comunidad pone especial atención en la separación como un aspecto fundamental a los fines de proporcionar seguridad de la información. Rushby y Randell fueron los primeros en introducir estos cuatro tipos principales de separación relacionada con la seguridad [Rushby 83], ante todo como un medio para separar entidades de diferentes niveles de clasificación. Estas estrategias también han demostrado ser de utilidad para la integridad y la disponibilidad de la información. La más vieja de las estrategias, la separación física, favorece la seguridad empleando mecanismos de distribución espacial y seguridad física [NCSC 88], tales como edificios reforzados, cerraduras, y diferentes tipos de escudos. La separación lógica emplea mecanismos basados en software, tales como filtros de mensajes, *wrappers* funcionales y *kernerls* seguros, para controlar el acceso. La separación criptográfica emplea la encriptación y la administración de claves para proteger la confidencialidad de los datos y para detectar la corrupción de los mismos en un grado proporcional a la robustez del algoritmo criptográfico y a la protección de las claves privadas. Finalmente, la separación temporal separa en el tiempo la ejecución de funciones críticas; es la que más próxima se encuentra asociada con los períodos de procesamiento, "el procesamiento de varios niveles de información sensible en momentos distinguiblemente diferentes" [NCSC 88].
- *Administración del Personal*. La capacidad de supervivencia de cualquier misión depende especialmente de la responsabilidad, conocimiento y capacidad de las personas a cargo del soporte o ejecución de la misión. La responsabilidad generalmente se tasa a través de procedimientos de seguridad del personal [NCSC 88] tales como una periódica investigación del nivel de conocimiento que poseen las personas que tienen a su cargo las responsabilidades de la misión. La evaluación del desempeño demostrado a me-

nudo son un complemento necesario de tal investigación y proporcionan información adicional en términos de comprensión y aptitud de un individuo para realizar adecuadamente la tarea. El entrenamiento periódico también es importante para educar a las personas sobre el rol que juegan sus trabajos para alcanzar exitosamente los objetivos de la misión, la importancia de la política de seguridad y los procedimientos asociados, y los posibles impactos de un desempeño inadecuado.

4.4.4 Refinamiento de la Arquitectura Conceptual



La arquitectura conceptual de un sistema evoluciona, como se describiera anteriormente, a través del análisis de amenazas y de la mitigación del riesgo utilizando tácticas de supervivencia. A través de esta evolución, TRIAD requiere la documentación de los requerimientos de supervivencia del sistema. La técnica adoptada en TRIAD para la especificación de los requerimientos de supervivencia deriva del uso de escenarios para la descripción y análisis de los requerimientos [Weidenhaupt 98] (que en la terminilogía UML, los escenarios se denominan casos de uso). Potts describe a los escenarios como una muestra de ejecución de un sistema, la antítesis de las especificaciones:

> *En tanto una especificación describe el comportamiento en general, un escenario ejemplifica el comportamiento mediante la presentación de episodios específicos y concretos. En tanto resulta posible deducir escenarios a partir de una especificación, sólo resulta posible inducir una especificación a partir de un conjunto de escenarios* [Potts 95].

Los requerimientos del sistema, tanto en los que hace a requerimientos funcionales como no-funcionales, se pueden definir como escenarios. Los requerimientos no-funcionales son requerimientos acerca de atributos de calidad tales como desempeño, usabilidad y mantenibilidad. Bass define una aproximación para caracterizar un atributo de calidad como un conjunto de escenarios generales [Bass 01]. *Un escenario general se describe en términos de un estímulo y una medida de la respuesta*; por ejemplo:

- Un escenario general de modificabilidad se estimula mediante arribo de cambios y da por resultado su propagación a través de la especificación e implementación del sistema; los escenarios generales de modificabilidad reflejan las diferentes clases de cambios posibles.
- Un escenario general de desempeño se estimula mediante el arribo de eventos y da por resultado una respuesta a los eventos con cierta latencia; los escenarios generales de desempeño reflejan las diferentes clases de desempeño de respuestas requeridos.

Bass propone que un conjunto de escenarios dependientes del sistema pueden servir para caracterizar completamente un atributo de calidad. Más aún, las especializaciones de escenarios generales, denominados escenarios específicos, se pueden utilizar para describir requerimientos no-funcionales dependientes del sistema. ***La noción de que un atributo de calidad pueda ser caracterizado como un conjunto de escenarios posee clara relevancia en el diseño de los sistemas con capacidad de supervivencia****. Un escenario general de supervivencia es estimulado mediante ataques perpetrados y da por resultado la resistencia, el reconocimiento, la recuperación y la adaptación de manera de continuar proveyendo los servicios esenciales. Los escenarios específicos de supervivencia reflejan las diversas clases de requerimientos para resistir, reconocer, recuperarse y adaptarse ante ataques.*

La elicitación de los requerimientos de supervivencia tiene que representar a aquéllos con responsabilidad sobre la misión, los usuarios eventuales del sistema y aquéllos con responsabilidad en lo que hace a la ejecución de la respuesta.

<table>
<tr><th colspan="2" rowspan="2">Estímulo</th><th colspan="5">Respuesta</th></tr>
<tr><th colspan="2">Resistencia</th><th>Reconocimiento</th><th colspan="2">Recuperación</th><th>Adaptación</th></tr>
<tr><td rowspan="2">Clase de ataque principal</td><td>Subclase #1 de la clase principal de ataque</td><td>Primera técnica para resistir ataques de la subclase #1</td><td>Segunda técnica para resistir ataques de la subclase #1</td><td rowspan="2">Técnica para reconocer ataques tanto de la subclase #1 como de la subclase #2</td><td rowspan="2">Técnica para recuperarse ante ataques tanto de la subclase #1 como de la subclase #2</td><td>Técnica adicional para recuperarse ante ataques de la subclase #1</td><td>Técnica para adaptarse ante ataques de la subclase #1</td></tr>
<tr><td>Subclase #2 de la clase principal de ataque</td><td colspan="2">Primera técnica para resistir ataques de la subclase #2</td><td>Técnica adicional para recuperarse ante ataques de la subclase #2</td><td>Técnica para adaptarse ante ataques de la subclase #2</td></tr>
</table>

Tabla 12. Formato tabular para los requerimientos de supervivencia.

Los requerimientos de supervivencia pueden estar convenientemente organizados dentro de una tabla, como se ejemplifica en la Tabla 12. La misma ilustra un *framework* para la especificación

de las diferentes maneras en que se requiere que responda un sistema frente a un estímulo de ataque. Una clase de ataque a ser atendida por el sistema se puede dividir en un cierto número de subclases. La tabla muestra de qué manera especificar las respuestas como una combinación de técnicas de resistencia, reconocimiento, recuperación y adaptación. Como se muestra, una respuesta individual puede atender a múltiples subclases de ataques. Asimismo, una subclase particular de ataque puede ser atendida por múltiples respuestas, aún dentro de la misma clase de técnicas de respuesta, como se muestra en el caso de resistencia para la subclase #1 de ataque. Esto proporciona una capacidad de especificar una defensa en-profundidad contra ataques particulares. El Anexo 3 provee ejemplos específicos de utilización de este formato tabular para la especificación de los requerimientos de supervivencia.

TRIAD generalmente arranca con una descripción del ambiente operacional esperado y con el soporte computacional deseado para los procesos de trabajo esenciales, tal como se los podría encontrar en el concepto de las operaciones. El ambiente operacional es refinado a lo largo del ciclo de vida del sistema. El alcance del análisis está influenciado por la naturaleza del ambiente operacional: la complejidad de las interacciones en el sistema, la distribución de los procesos de trabajo, las dependencias trabajo-proceso, y la compartición de riesgos de supervivencia y responsabilidades a lo largo de múltiples organizaciones. Los escenarios de supervivencia deberían describir la respuesta del sistema tanto para los estímulos esperados y los no esperados. Los ataques exitosos sobre los sistemas a menudo resultan de estímulos que se encuentra fuera del rango de aquéllos esperados por los diseñadores del sistema. El comportamiento del sistema cuando es confrontado con tales estímulos es, en consecuencia, muy importante para el establecimiento de la supervivencia del sistema. La arquitectura conceptual debe, finalmente, documentar la respuesta del sistema ante estímulos esperados y no esperados.

Las tácticas de supervivencia sirven como los bloques constructivos durante el diseño de la arquitectura del sistema para la satisfacción de escenarios específicos de supervivencia. Esto es análogo a la manera como Bachmann utiliza las tácticas arquitectónicas para satisfacer los atributos de calidad [Bachmann 02]. Por ejemplo:

- *Encapsulamiento* es una táctica arquitectónica destinada principalmente a mejorar la modificabilidad limitando la propagación del efecto de los cambios.
- *Replicación* es una táctica arquitectónica destinada a mejorar el desempeño mediante la reducción del tiempo de respuesta a través de la localización o la mejora de la fiabilidad proveyendo copias redundantes de funciones o datos.

TRIAD introduce tácticas de supervivencia dentro de la arquitectura de manera iterativa, para atender ataques dirigidos a elementos diferentes y que requieren de crecientes grados de sofisticación del atacante. Al igual que con otros atributos de calidad, estas tácticas sirven para

satisfacer escenarios específicos de supervivencia, lo cual caracteriza la supervivencia para su aplicación.

En el Anexo 3 se presenta un ejemplo de aplicación del modelo TRIAD.

SECCION 5: ¿SE PODRÁN CONSTRUIR SISTEMAS CON CAPACIDAD DE SUPERVIVENCIA EMPLEANDO COMPONENTES COTS?

Como se planteara en la Propuesta de Tesis[15], se ha tomado como base el Plan de Trabajo del CERT en el campo de los Sistemas con Capacidad de Supervivencia. En dicha propuesta, se expuso que:

> *Se requieren de métodos de evaluación sistemáticos para establecer la supervivencia de un componente COTS* (*commercial off-the-shelf*). *Muchas organizaciones se encuentran desarrollando sistemas de misión crítica en los que emplean componentes COTSs. Los COTSs suelen presentar menores costos que las soluciones hechas a medida, pero las organizaciones que los adquieren carecen de acceso a los artefactos del proceso de ingeniería de software utilizados para crear los componentes.*

Por ello, resulta necesario abrir el espectro de análisis a trabajos relacionados con el tema central. En particular, se incluye el reporte realizado por Howard F. Lipson, Nancy R. Mead y Andrew P. Moore publicado en Diciembre de 2001, titulado "*Can we ever build Survivable Systems from COTS components*". Este equipo de trabajo ha desarrollado una solución orientada a la gestión del riesgo, denominada *Vendor Risk Assesment and Threat Evaluation – V-RATE*, destinada a la evaluación de los sistemas basados en COTSs.

Se espera que esta herramienta ayude a las organizaciones compradoras a comprender el equilibrio que debe alcanzarse entre el uso de productos COTSs y la posibilidad de alcanzar los niveles de seguridad requeridos mediante la evaluación de diferentes diseños de sistemas basados en estos productos.

El empleo componentes COTS para la construcción de sistemas extensos y complejos se ha vuelto una manera corriente en que este tipo de sistemas se diseñan e implementan por parte de los organismos gubernamentales y las empresas. Gran parte de la literatura referida a sistemas basados en COTS reconoce que tales sistemas no son adecuados para las aplicaciones de misión crítica. Sin embargo, existe una considerable evidencia de que los sistemas basados en COTS se han venido utilizando en dominios en los que resultan posibles el daño económico e incluso las pérdidas de vida ante el evento de una falla o compromiso importantes del sistema. ¿Se pueden construir estos tipos de sistemas de tal manera que los riesgos se correspondan con los que por lo general se toman en otras áreas de la vida o de los negocios?

Este trabajo describe un *framework* de mitigación de riesgos para la toma de decisión acerca de

[15] Propuesta de Tesis Maestría en Redes de Datos "Análisis de *Survivable Networks* y Evaluación de la Metodología TRIAD", Postulante Ing. Susana C. Romaniz.

cuándo y cómo se pueden utilizar componentes COTS para construir sistemas con capacidad de supervivencia. La exitosa aplicación del mismo requiere de un trabajo conjunto con los proveedores a fin de reducir los riesgos asociados con el empleo de sus productos, y el mejoramiento y el mejor uso de la experticia en cuando a gestión del riesgo de la propia organización.

Los bajos costos, y la creencia de que la reducción de gastos extiende el ciclo de vida del sistema son las principales motivaciones al momento de optar entre sistemas a-medida y basados en COTS. Las desventajas asociadas con el diseño de sistemas basados en COTS incluye la ausencia del código fuente y la falta de acceso a los artefactos del proceso de desarrollo del software utilizado para el diseño de los componentes COTS.

Aún cuando se vaya a construir un sistema utilizando componentes COTS de diferentes marcas, o que haya sido un único proveedor que ha entregado una solución integrada, muchos de los riesgos asociados con la administración y la operación del sistema están fuera de control directo [Basili 01, Brownsword 00, Hissam 98, Lindqvist 98]. *Cada vendedor que juega un rol en el diseño, desarrollo, adquisición, integración, despliegue, mantenimiento, operación o evolución de parte (o todo) del sistema afecta los riesgos ante el hecho de intentar sobrevivir a ciberataques, accidentes y fallas de sub-sistemas. Se propone la realización una continua evaluación de riesgo basado en proveedor como una parte crítica del ciclo de vida del sistema para el caso de los sistemas de misión-crítica que emplean componentes COTS*.

Los sistemas con capacidad de supervivencia son aquellos que continúan satisfaciendo sus misiones (tal vez a nivel reducido de servicio), a pesar de tener componentes o sub-sistemas dañados o comprometidos por el ataque, el accidente o la falla. Las investigaciones llevadas a cabo en el *Software Engineering Institute* han demostrado que la supervivencia de un sistema es dependiente de cuán bien se hayan equilibrado los diferentes atributos de calidad de la arquitectura e implementación de un sistema. El equilibrio entre diseño racional y los atributos de calidad se da entre los muchos artefactos de ingeniería que no están disponibles a los compradores de componentes COTS. Entonces, ¿resulta imposible construir sistemas con capacidad de supervivencia que no incluyan COTS?

La gestión del riesgo resulta central para alcanzar la supervivencia [Lipson 99]. *Aquéllos que adquieran, diseñen, implementen, operen, mantengan y evolucionen sistemas que utilizan componentes COTS pueden mejorar significativamente la supervivencia de dichos sistemas trabajando con sus proveedores a fin de reducir los riesgos inherentes existentes en los productos y procesos de dichos componentes, y mediante el mejoramiento y haciendo el mejor uso de la experticia en cuanto a la gestión de riesgos que posea la organización*. Este trabajo sugiere mecanismos que alientan el desarrollo de componentes COTS futuros y procesos del proveedor que ofrezcan la suficiente visibilidad sobre los aspectos intrínsecos del producto a fin de propor-

cionar evidencia del empleo de estos componentes puede contribuir al aseguramiento de la supervivencia global del sistema.

5.1 SUPERVIVENCIA Y COMPONENTES COTS

No es posible alcanzar la supervivencia sin una clara comprensión del contexto en el que los sistemas modernos operan generalmente –dominios ilimitados. Este tipo de dominios, tal como la Internet, están caracterizados por la falta de un control central y la falta de información completa o precisa en tiempo y forma. Más aún, un sistema típico hoy en día constituye un dominio ilimitado. Ante la ausencia de un control completo y de una visibilidad completa sobre el sistema y su ambiente, alcanzar la supervivencia (es decir, satisfacer la misión de un sistema) es un ejercicio en el campo de gestión de riesgos y tolerancia. ***Si el sistema está compuesto principalmente de componentes COTS, se está casi en una situación extrema en lo que hace a la falta de control y visibilidad respecto del comportamiento último del sistema bajo una variedad de circunstancias que podrían amenazar su supervivencia*** [Mead 01].

Las vulnerabilidades del sistema que son extremadamente improbables de provocar la falla de la misión debido a las acciones de un usuario normal, sí pueden muy probablemente ser explotadas por un adversario inteligente (por ejemplo, aprovechando vulnerabilidades de *buffer-overflow*. Esto puede resultar posible particularmente cuando se dispone de *scripts* en los que se ha codificado los a menudo intrincados y detallados pasos requeridos para la exitosa explotación. En consecuencia, la supervivencia demanda que se satisfagan con alto grado de aseguramiento los siguientes requerimientos:

- Tales vulnerabilidades no existen o no pueden ser explotadas.
- Si tales vulnerabilidades pueden ser explotadas, su explotación no debería comprometer la misión o debería ser reconocida y recuperarse de la misma en tiempo como para continuar la misión [Ellison 99a].

Esta necesidad de un alto aseguramiento es lo que hace tan dificultoso el empleo de componentes COTS en sistemas de misión crítica.

Desde luego que los componentes COTS siempre se pueden utilizar para la implementación de funciones no-críticas de un sistema, es decir, funciones cuya propiedades no impacten en la supervivencia de un sistema. Algunas arquitecturas han demostrado cómo estructurar un sistema de tal manera que la función crítica se encuentre aislada en un conjunto pequeño de componentes de alto aseguramiento, permitiendo así el empleo de componentes COTS en cualquier otra parte [Froscher 98].

Allí donde tales soluciones no son posibles de aplicar, la pregunta pendiente de contestar en

si los componentes COTS pueden ser utilizados para implementar funciones críticas de sistemas. Las técnicas de supervivencia a menudo descansan en la redundancia para tolerar los compromisos de componentes individuales. Las defensas en capas (por ejemplo, emplear detección y recuperación de intrusión para complementar las medidas de resistencia) también colaboran en la tolerancia a fallas a la hora de implementar una función crítica. *Como resultado de la prudencia del equipo de diseño, el empleo de replicación, redundancia y diversidad, la adecuación de un sistema con capacidad de supervivencia puede emerger de las interacciones entre los componentes particulares de un sistema aún cuando los componentes en sí mismos no posean dicha capacidad.* ***Pero, ¿qué garantías de aseguramiento se le deben exigir a los componentes COTS que implementan las funciones críticas de sistema?***

La criticidad de un sistema influye en los requerimientos de aseguramiento aplicados a los componentes COTS que implementan servicios esenciales. Un sistema posee alta criticidad si las consecuencias de una falla del sistema son severas. Un sistema posee baja criticidad si las consecuencias de la falla del sistema son insignificantes. La Figura 28 asocia el aseguramiento requerido a los componentes COTS como una función de la criticidad del sistema. Existen muchos factores que influyen en el aseguramiento de un componente COTS, los cuales serán analizados posteriormente. No obstante, a fin de analizar la figura, se ha considerado dicho aseguramiento de una manera abstracta, asumiendo solamente que el aseguramiento puede variar significativamente.

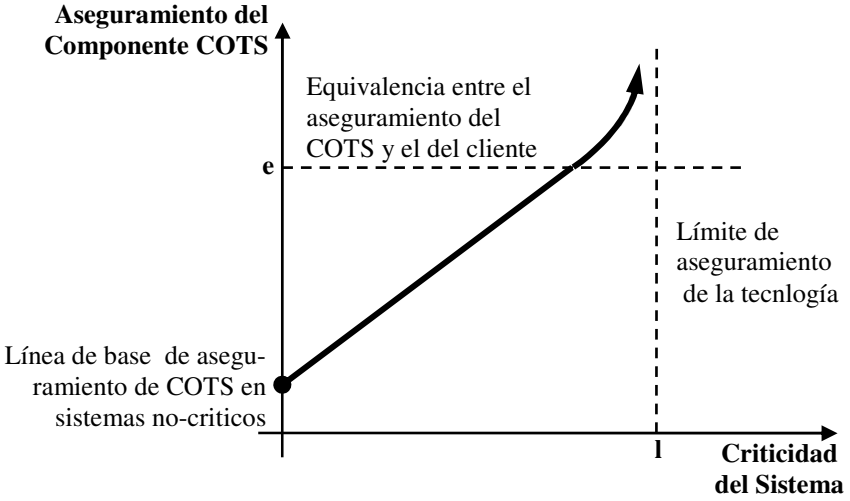


Figura 28. Aseguramiento requerido a un componente COTS como función de la Criticidad del Sistema

Se ha asumido en Figura 28 que existe alguna línea base en lo que se refiere al aseguramiento de un componente COTS que se requiere aún para sistemas no-críticos. Suponiendo que (c,a) es un punto particular perteneciente a la curva, donde $c < l$ es la criticidad del sistema en construcción, entonces a representa el aseguramiento mínimo permitido de cualquier componente COTS

utilizado para implementar los servicios esenciales del sistema. El área por encima de la curva representa un uso aceptable de los componentes COTS, en tanto que el área por debajo de la curva representa un uso inaceptable. Se ha asumido que el sistema propuesto descansa en el uso de componentes COTS para la supervivencia de la misión del sistema. También se ha asumido que existe un punto para el cual el impacto potencial es tan severo que la tecnología no debería ser utilizada a pesar del aseguramiento que la tecnología proporcione. Este límite es la línea de punto *l*. La naturaleza asintótica de la curva con respecto al límite *l* refleja la necesidad de un aseguramiento "infinito" por parte del componente COTS para la implementación de tales sistemas de alto-aseguramiento.

La línea de puntos etiquetada con *e* representa el aseguramiento del componente COTS que resulta tan convincente como el de uno desarrollado a medida. En tanto esto varía con el proceso empleado en el desarrollo a medida, la postulación de tal punto refleja la posibilidad real de que el aseguramiento del componente COTS pueda exceder el del desarrollado a medida. Se puede argumentar que componentes COTS con tal alto aseguramiento no existen actualmente y, aún en el caso que así sea, en esa situación no pueden ser considerados como componentes COTS. No obstante, se considera que dentro de la caracterización de COTS que se realiza en el presente trabajo, éstos pueden contar con ciertos beneficios en lo que hace al aseguramiento por encima de los desarrollados a medida; por ejemplo, la experticia de fabricante, los antecedentes de uso, y la garantía de evolución. Así las cosas, la parte de la cursa por encima de la línea *e* representa los sistemas que son tan críticos que realmente requieren de componentes COTS de muy alto aseguramiento en lugar de un desarrollo a medida.

5.2 COTS VS. DISEÑO A MEDIDA ¿UNA OPCION BINARIA O UNA VARIEDAD DE OPCIONES?

El término COTS, como generalmente se lo entiende, hace referencia a una amplia variedad de software producido comercialmente provisto como código objeto, respecto del cual la única información que se puede contar es el precio de compra, un listado de características, un manual de usuario, algunas afirmaciones acerca del producto, un acuerdo de licencia, una especificación de API, y la experiencia que se haya tenido con el producto o con alguna versión de ensayo (o la experiencia de algún tercero o resultados de pruebas de las cuales se tenga conocimiento). En consecuencia, la visibilidad que se posee del producto y del proceso utilizado para su construcción se encuentra extremadamente limitada. El único control que se tiene sobre el producto es si adquirirlo o no y cuántas copias se han de adquirir.

En el extremo opuesto del mundo del desarrollo de software están el software y los sistemas desarrollados a medida. Sin embargo, el diseño y el desarrollo de software siempre comportan la

gestión de riesgos, debido a que aún en el caso en que se empleen procesos de software de alta calidad siempre está presente la falta de un completo control y de una completa visibilidad. Las metodologías de análisis y diseño iterativos pueden ayudar a determinar los atributos funcionales y no-funcionales definitivos de un sistema; y el equilibrio entre atributos de calidad, funcionalidad y costos del software se alcanza cuando se ha estimado conveniente que los riesgos asociados con el diseño, implementación, implantación y uso del sistema se encuentran por debajo de el umbral de tolerancia al riesgo [Kazman 98]. Más aún, muy probablemente el compilador y otras herramientas de programación son productos COTS, al igual que las plataformas de desarrollo y de producción. Nunca se puede tener un conocimiento total acerca de la formación y habilidades del personal encargado de la construcción del sistema, a pesar de los esfuerzos que se hagan en términos de chequear las referencias a fin de reducir el riesgo, a costa de algún tiempo y gastos adicionales.

En consecuencia, resulta necesario disipar la idea de que las organizaciones que consideran que ante la necesidad de construir sistemas con capacidad de supervivencia sólo se encuentran a una opción binaria, COTS o a medida, en la que los sistemas basados en COTS carecen de casi todo control y visibilidad, pero que son relativamente más económicos, y que los sistemas a medida ofrecen un control y una visibilidad totales, aunque a un costo mucho mayor. *Existe una variedad de opciones de diseño, que van desde un 100% de integración de "cajas negras" COTS hasta un 100% de integración de desarrollos a medida, que permite una solución más flexible, a un costo eficiente y de riesgos aceptables para el diseño de sistemas con capacidad de supervivencia*.

El Método V-RATE "*Vendor Risk Assessment and Threat Evaluation*" que se describe a continuación, delinea un conjunto de estrategias para la mitigación de los riesgos asociados con el empleo de productos COTS. Algunas de las estrategias posibilitan una reducción de los riesgos proveyendo control y visibilidad sobre los aspectos internos de un producto y los procesos utilizados para su construcción. No obstante, el Método V-RATE incluye otras alternativas para la mitigación del riesgo que no incluyen un incremento en el control y la visibilidad.

5.3 EL METODO V-RATE

La construcción de sistemas con capacidad de supervivencia empleando componentes COTS es una tarea intimidante debido a que el desarrollador posee escaso o ningún acceso a los artefactos del proceso de ingeniería de software utilizado para crear los componentes. Estos artefactos son las principales fuentes a partir de las que se tienen evidencias del aseguramiento que presentará sistema compuesto que se derivado. *Una manera de compensar parcialmente es emplear la*

valorización del riesgo del fabricante como una herramienta que ayude a construir, mantener y evolucionar sistemas con capacidad de supervivencia. Este tipo de valorización se puede emplear como una nueva fuente de evidencia de aseguramiento de la supervivencia de un sistema.

La valorización propuesta se basa en la taxonomía de valorización del riesgo del fabricante y evaluación de la amenaza. En el nivel más alto de la taxonomía se tienen dos grandes categorías: (1) ***elementos de riesgo inherentes al fabricante***, y (2) ***elementos de riesgo del fabricante que están asociados con la experticia en la gestión de riesgos de la organización***. El resultado de una valorización basada en esta taxonomía es un ***perfil de riesgo del fabricante*** correspondiente al sistema que está en evaluación. Resulta esperable contar con una extensa y creciente colección de perfiles ligados a historias de desempeño en el mundo real, que provean datos empíricos contra los cuales poder comparar un perfil de riesgo nuevo. *Un perfil de riesgo del fabricante se puede utilizar para valorizar el riesgo asociado con el uso de un producto en un ambiente de amenaza y para identificar las áreas de actividades de mitigación del riesgo adicionales. Debido a que un único valor promedio no proporcionará la guía suficiente para estas actividades de mitigación del riesgo, este perfil ayuda a identificar los riesgos presentes en cada área de la taxonomía y permite que se evalúe la tolerancia al riesgo existente con respecto a cada elemento de la taxonomía.*

5.3.1 Taxonomía V-RATE

Los elementos que se incluyen en la taxonomía son:

1. Elementos de Riesgo Inherentes al Fabricante

1.1 Visibilidad de los Atributos del Producto

1.1.1 Grado de apertura de la visibilidad en los procesos de diseño y desarrollo

1.1.2 Testeo por parte de organizaciones independientes

1.2 Aptitud Técnica

1.2.1 Madurez de la capacidad de supervivencia

1.2.2 Existencia de certificaciones y calificaciones del fabricante

1.2.3 Evidencia de adhesión a los estándares aplicables y a regulaciones gubernamentales

1.2.4 Diversidad y redundancia demostradas en productos y servicios del fabricante

1.2.5 Existencia de un equipo del fabricante que maneje de manera efectiva cuestiones relativas a la seguridad y a la supervivencia

1.3 Historia de Desempeño

1.4 Conformidad

1.4.1 Receptividad a cuestiones relacionadas con la seguridad y la supervivencia (lo que puede incluir aspectos relativos con al calidad tales como fiabilidad, des-

empeño, protección y usabilidad)

1.4.2 Receptividad frente a solicitudes de mejoras y de características nuevas

1.4.3 Disposición a cooperar con testeadores y certificadores de terceras partes

1.5 Fidelidad

1.5.1 Seguimiento de conversaciones y de registros

1.5.2 Evidencia de la experticia en la evaluación de la fidelidad del personal

1.6 Competencia en la Gestión del Negocio

1.6.1 Viabilidad económica

1.6.2 Experticia en gestión de riesgos del fabricante en el trato con sub-contratistas

1.7 Evolución Controlada

1.7.1 Camino de evolución especificado de manera clara (o discernible)

1.7.2 Estabilidad en la integración de productos

1.7.3 Soporte en la evolución del producto en cuanto mejoras continuas relacionadas con la supervivencia

2. Elementos de Riesgo del Fabricante Asociados con la Experticia del Especialista en la Gestión del Riesgo con Fabricantes

2.1 Factores Técnicos de Mitigación del Riesgo

2.1.1 La experticia del especialista en la evaluación de los atributos de calidad de producto (en particular, aquellos atributos de calidad que pueden contribuir con la supervivencia del sistema, tales como seguridad, fiabilidad, desempeño, protección y usabilidad)

2.1.2 La experticia del especialista en evaluar la competencia técnica del fabricante

2.1.3 Conocimiento de la existencia de calificaciones y certificaciones de evaluación del fabricante

2.1.4 Diversidad y redundancia demostradas en la integración de productos y servicios de fabricante

2.1.5 Empleo de herramientas y técnicas a nivel de la arquitectura (por ejemplo, *wrappers*) para limitar los riesgos asociados con un producto particular del fabricante

2.1.6 Asociación del especialista con organizaciones relacionadas con la seguridad y la supervivencia y la existencia de un grupo dedicado a la seguridad y la supervivencia dentro de la organización

2.2 Mitigación No-Técnico del Riesgo

2.2.1 Legal

2.2.2 Económico

2.2.3 Político y social

2.3 Independencia / Interdependencia

2.4 Grado de Exposición de la Organización

2.5 Alineamiento con la Misión / Compatibilidad del Fabricante

2.6 Experticia en la Negociación / Poder de Negociación

5.3.2 Técnicas específicas de reducción del riesgo del fabricante

El Método V-RATE proporciona un *framework* para la valorización de los riesgos de supervivencia asociados con productos COTS. Si bien existen muchos riesgos y mucho trabajo por hacer, se cuenta con técnicas específicas en las que dichos riesgos se pueden reducir. En el largo plazo, se debería contar con una lista de técnicas de reducción del riesgo del vendedor. Cada técnica debería tener asignada un valor que se podría emplear en el cálculo del V-RATE para demostrar la reducción del riesgo global de supervivencia asociado con productos COTS específicos.

Para cada elemento de la taxonomía V-RATE, se deberían desarrollar estrategias de reducción del riesgo. En la Tabla 13 se proponen ejemplos breves de medios con los cuales se pueden reducir los riesgos; dichos ejemplos están alineados con la taxonomía V-RATE.

A continuación se desarrollan ejemplos de dos ítems de la tabla; se los podría desarrollar para toda la tabla a fin de conformar un conjunto integral de ejemplos/estrategias.

5.3.2.1 Ejemplo de la taxonomía V-RATE de la Sección 1.4 Conformidad

El fabricante está deseoso de responder a inquietudes referidas a seguridad y supervivencia mediante:

- El desarrollo en forma inmediata parches de seguridad
- Permitiéndole el cliente que desactive características innecesarias y así reducir los riesgos asociados con éstas; de esta manera, el cliente puede seleccionar un núcleo de servicios requeridos, en lugar de verse forzado a convivir con las consecuencia de implementaciones generales y mediocres
- Construyendo mecanismos de recuperación embebidos en el software; ejemplos de estos mecanismos incluyen el respaldo automáticos de datos y retención del estado de los datos.
- Construyendo mecanismos de seguridad (resistencia) embebidos en el software; ejemplos de estos mecanismos son encriptación, protección de contraseñas, y diversidad.
- Implementando prácticas de ingeniería del software a fin de mejorar la seguridad, tales como inspecciones, testeos, empleo de lenguajes fuertemente tipados, y procesos que

soporten buenas prácticas de programación. Otra respuesta positiva por parte en lo que al cliente concierne sería iniciar o mejorar la capacitación en temas de seguridad e ingeniería de software del equipo técnicos del fabricante.

Elemento V-RATE	Ejemplo
1.1 Visibilidad de los atributos del producto	El fabricante está deseoso de permitirle al cliente tener acceso al código fuente correspondientes a los binarios instalados
1.2 Competencia técnica	El fabricante ha demostrado competencia en áreas claves relacionadas con la supervivencia (utilizando un modelo de madurez de la capacidad de supervivencia)
1.3 Historia de desempeño	El fabricante cuenta con un registro de seguimiento – experiencia, estadísticas, testimonios y referencias verbales
1.4 Conformidad	El fabricante desarrolla en forma inmediata parches de supervivencia
1.5 Responsabilidad	El fabricante chequea en forma consistente la naturaleza de las referencias de los nuevos empleados y periódicamente vuelve a chequear las de todo el personal
1.6 Competencia en la gestión del negocio	Las perspectivas de sanidad económica en el largo plazo del fabricante son buenas
1.7 Evolución controlada	El fabricante comparte planes y procedimientos que indican la evolución controlada del producto
2.1 Factores técnicos de mitigación del riesgo	Se cuenta con la experticia necesaria para la evaluación técnica del riesgo en forma directa (incluida, pero no limitado, la aplicación del Método SNA)
2.2 Mitigación no-técnica del riesgo	Se tiene acceso a protección legal o económica, tal como seguros, acuerdos de garantía y licenciamiento, cláusulas de desempeño y penalidades asociadas, protección de la regulación, y límites de desempeño
2.3 Independencia / Interdependencia	Se examinan los productos y servicios del fabricante asociados con el sistema en estudio y se buscan interdependencia que podrían amenazar la supervivencia
2.4 Grado de exposición	Se determinan cuáles elementos del sistema son dependientes de la competencia, la fidelidad y la minuciosidad del fabricante
2.5 Alineamiento con la misión / Compatibilidad del fabricante	Se evalúa el alineamiento con la misión de la organización y los atributos de calidad del software (SQAs) con la misión y los SQAs del fabricante
2.6 Experticia en la negociación / Poder de negociación	Comunidad de intereses con el fabricante para obtener la notificación temprana de potenciales problemas de seguridad y supervivencia

Tabla 13. Ejemplos de reducción de riesgos V-RATE.

5.3.2.2 Ejemplo de la taxonomía V-RATE de la Sección 1.7 Evolución Controlada

Los planes y procedimientos del fabricante indican la evolución controlada del producto de la siguiente manera:

- Las actualizaciones del fabricante no requieren de una re-integración masiva, tales como la re-escritura del código de APIs.
- La aplicación de parches de seguridad no se debería ver demorada por los efectos de pendulares en los cambios en el producto del vendedor.
- Existe un bajo grado de acoplamiento de características
- Cambios en algunas pocas características no son causa de complejas tareas de mantenimiento masivo
- El fabricante está deseoso de comunicar los planes de negocio relacionados con el producto, de tal manera que el cliente posee alguna idea de la estabilidad del producto
- El fabricante acepta dar soporte del producto, en particular desde una perspectiva de la seguridad y la supervivencia, en el largo plazo

En el Anexo 4 se presenta un ejemplo de aplicación del método V-RATE.

5.4 DE QUÉ MANERA EL METODO V-RATE SE RELACIONA CON EL *COMMON CRITERIA*

El estándar internacional ISO 15408, denominado *Common Criteria* (CC), representa un intento de contar con una solución estructurada y flexible que ayude a los consumidores y fabricantes de productos COTS relevantes a acordar y evaluar las funciones requeridas de un producto y el aseguramiento tanto del producto como del proceso [CCIMB 99]. El CC promueve la creación de dos documentos denominados "Perfil de Protección" (*Protection Profile*) y "Objetivo de Seguridad" (*Security Target*). Los consumidores desarrollan un Perfil de Protección para una clase de productos de seguridad de su interés, tal como *firewalls*, sistemas operativos, y *smart cards*. El Perfil de Protección especifica la función y los aseguramientos requeridos por una amplia base de fabricantes del producto independiente de cualquier implementación particular. Un fabricante desarrolla un Objetivo de Protección para describir su implementación de un producto destinado a conformar un Perfil de Protección particular. El Objetivo de Seguridad especifica las funciones soportadas, el proceso de desarrollo utilizado, y un argumento por el cual las funciones y los procesos satisfacen un Perfil de Protección particular. El CC establece una guía para la producción y la evaluación independiente de un Objetivo de Seguridad de un fabricante que responde a un Perfil de Protección de un consumidor.

Si bien originalmente estuvo pensado como un medio para la evaluación de información de tecnología de la seguridad aceptado internacionalmente, el CC puede proporcionar un modelo para emplear el método V-RATE y así estimular la creciente credibilidad de los productos COTS. V-RATE proporciona criterios para los productos y procesos que asisten en la evalua-

ción de la tecnología COTS a ser utilizada para alcanzar una misión particular. Los criterios de V-RATE son independientes de la tecnología y, en consecuencia, son más abstractos que el CC. El método V-RATE también incluye criterios para valorizar la propia habilidad del consumidor para tratar con fabricantes COTS y los riesgos inherentes asociados con la tecnología COTS. Una de las críticas que se le hace al CC es la gran esfuerzo que se requiere para producir y evaluar productos dentro de framework. V-RATE puede proporcionar un punto intermedio entre la aceptación de COTS a caja negra y de un producto evaluado vía CC.

Una diferencia significativa entre el método CC y el V-RATE es que éste es conducido por la parte cuya seguridad y supervivencia se encuentra en riesgo, mientras que la evaluación CC generalmente es pagada por el fabricante y conducida en nombre del mismo [Anderson 01].

5.5 RESUMEN Y TRABAJO FUTURO

Muchas organizaciones tienen una postura de "todo o nada" con respecto al uso de componentes COTS en los sistemas de misión crítica (por ejemplo, o los componentes COTS nunca son seguros de utilizar, o se debería maximizar el empleo de COTS). Este trabajo describe los criterios V-RATE que asisten en la toma de decisiones respecto de cuándo y cómo se pueden utilizar productos COTS para construir sistemas con capacidad de supervivencia. Los factores que influyen en esta decisión incluyen no sólo atributos de los productos COTS en sí mismos, sino también atributos de la misión del sistema, del fabricante, del proceso de ciclo de vida de desarrollo del fabricante, y de las habilidades de gestión del riesgo de la propia organización.

Una mayor cooperación del fabricante mejora la efectividad del método V-RAGTE. A menudo, las organizaciones esperan muy poco de los fabricantes, en términos de visibilidad de los aspectos internos de sus productos y procesos, o de la significación de las garantías de calidad. Estas expectativas necesitan ser mejoradas de tal manera que los fabricantes respalden de manera más directa los esfuerzos de valorización y reducción del riesgo de sus clientes. Más aún, se necesitan explorar los incentivos económicos que alienten la cooperación del fabricante.

En el futuro se continuará investigando de qué manera posicionar al método V-RATE sobre bases más científicas. Su actual desarrollo puede ofrecer una entrada a un modelo similar al *Capability Maturity Model* -CMM- que podría ayudar a los compradores en un proceso sistemático de valorización de la madurez del fabricante en la producción de componentes COTS para sistemas con capacidad de supervivencia. Fundamentos más rigurosos requerirán de mediciones cuantitativas de la capacidad de un sistema de sobrevivir ante ataques maliciosos, de medios para la medición de la medida en que un producto (o conjunto) COTS dado favorecen u obstruyen esa capacidad. Esto debe incluir la habilidad de medir el impacto sobre la supervivencia del

sistema de las interacciones entre múltiples componentes COTS.

También se espera incorporar los criterios V-RATE en los modelos de ciclo de vida de un sistema, tal como el Modelo en Espiral. El proceso de refinamiento de una arquitectura con capacidad de supervivencia que emplea productos COTS es inherentemente iterativo. Los riesgos inaceptables de un producto o de un fabricante que se detecten durante una iteración pueden hacer que se retroceda a una iteración previa a fin de incorporar en la arquitectura productos diferentes o de distinto fabricante. Esta naturaleza iterativa manejada por riesgo del modelo en espiral resulta particularmente apropiada para la incorporación del método V-RATE.

Este plan de incorporar los criterios V-RATE dentro de los modelos de ciclo de vida de desarrollo de software requerirá de poner una especial atención en el concepto de software de código abierto (*open source*) que está ganando una creciente aceptación en estos últimos años. El software de código abierto ofrece acceso al código fuente de un producto a reducir o ningún costo, con lo que alienta a la comunidad de desarrolladores a la lectura, modificación y redistribución del código fuente, conduciendo potencialmente a una rápida evolución y mejoramiento. Se ha de investigar la adecuación de los componentes de código abierto en el diseño, desarrollo, mantenimiento y evolución de los sistemas con capacidad de supervivencia, dentro del contexto de los criterios V-RATE y su integración con los componentes COTS más tradicionales.

Finalmente, resulta necesario aplicar el método V-RATE a los sistemas de misión crítica del mundo real. Estos casos de estudio han de ayudar a un ajuste fino y validación del método, y a demostrar su uso dentro de un proceso de ciclo de vida más realista. Estos estudios también ayudarán a comprender los riesgos asociados con los componentes COTS correspondientes a misiones específicas del sistema. Los detalles de la aplicación de V-RATE (tales como evidencia específica que necesita ser reunida) pueden diferir entre los distintos dominios (por ejemplo, sistemas financieros, de *e-commerce* o militares de misión crítica). Debido a que la capacidad de supervivencia es fuertemente dependiente del contexto de la misión, es fundamental comprender estas diferencias para la exitosa aplicación de V-RATE.

Se pretende focalizar las actividades de investigación en estas áreas y alentar a otros equipos a hacer algo similar.

SECCION 6: UN FRAMEWORK INTEGRAL DE ANALISIS Y DISEÑO: LA TECNOLOGÍA FSQ

Continuando con lo planteado en la Propuesta de Tesis[16], y tomando como base el Plan de Trabajo del CERT en el campo de los Sistemas con Capacidad de Supervivencia., se expuso que:

> *Las complejidades de un sistema distribuido de gran escala se pueden reducir y administrar mediante una disciplina de ingeniería unificada destinada al análisis y el diseño, que incluya la capacidad de supervivencia dentro de un framework integral. Las complejidades en el análisis y diseño de este tipo de sistemas a menudo exceden las capacidades de la ingeniería en lo que se refiere a su control intelectual.*

Hacia mediados del año 2002, el equipo formado por R. C. Linger, M. C. Pleszkoch, G. Walton, y A. R. Hevner publica el trabajo denominado "*Flow-Service-Quality (FSQ) Engineering: Foundations for Network System Analysis and Development*", en el que se presentan los fundamentos de la tecnología *Flow-Service-Quality – FSQ* [Hevner 02], la que está *basada en estructuras de flujo de tareas de usuario y en su rastreabilidad a través de la arquitectura, una aproximación computacional para calificar atributos (incluida la supervivencia), y un framework arquitectónico para la administración dinámica de los flujos y de sus atributos de calidad.*

Este proceso se puede aplicar tanto a la especificación, diseño y operación de sistemas nuevos, como al análisis de sistemas existentes en lo referente a dependencias y riesgos de supervivencia que puedan impactar en el desempeño de la misión. El mismo también ayuda a la integración de sistemas stovepipe[17] pre-existentes para soporte de nuevos objetivos de misión.

Dado que la sociedad moderna difícilmente podría funcionar sin los sistemas de información de gran escala centrados en redes que impregnan las organizaciones, las fallas o los compromisos graves en dichos sistemas desencadenan consecuencias de gran repercusión. Los actuales sistemas de información se caracterizan por su cambiante y a menudo desconocidas fronteras y componentes, su constante variación de funciones y utilización, y sus complejas de operaciones asincrónicas. *Su complejidad representa un reto al control intelectual por parte de personas, y su supervivencia se ha tornado una prioridad urgente. Se requiere de métodos de ingeniería basados en sólidos principios y de la realidad de los sistemas en red para gestionar la complejidad y asegurar la supervivencia.* ***La Ingeniería de Flujo-Servicio-Calidad (FSQ) es una tecnología emergente para la gestión, adquisición, análisis, desarrollo, evolución y operación de sistemas de gran escala, centrados en red. Está basada en Estructuras de Flujo, Atributos de***

[16] Propuesta de Tesis Maestría en Redes de Datos "Análisis de *Survivable Networks* y Evaluación de la Metodología TRIAD", Postulante Ing. Susana C. Romaniz.
[17] Denominación que se le da a aplicaciones centradas en un conjunto reducido de problemas.

Calidad Computables, y Arquitecturas de Gestión de Flujo. Estas tecnologías pueden ayudar a proporcionar principios de ingeniería estables para el dinámico e impredecible mundo de los sistemas de gran escala centrados en red*. Los principios FSQ están definidos como teoremas que iluminan las prácticas de ingeniería y las oportunidades de automatización*.

Las *Estructuras de Flujo* definen los flujos de tareas asociados con la misión de la organización y sus refinamientos derivados de los usos de los servicios de sistema que atraviesan la red. Los flujos son determinísticos para la comprensión humana, a pesar del asincronismo subyacente de las operaciones de red. Se los puede definir, abstraer y verificar con precisión, y tratar explícitamente con Factores de Incertidumbre, que incluyen funcionalidades inciertas de los COTS, y las fallas y compromisos del sistema.

Los *Atributos de Calidad Computables* van más allá de de las estimaciones a-priori y estáticas para tratar con atributos de calidad tales como fiabilidad y supervivencia como funciones dinámicas que se computan durante la operación del sistema. Los requerimientos de Atributos de Calidad Computables están asociados con los flujos y pueden ser dinámicamente reconciliados con los atributos del servicio de red durante la ejecución.

Las *Arquitecturas de Gestión del Flujo* incluyen el diseño y la implementación de marcos estructurales para la gestión dinámica de flujos y de requerimientos de atributos, al igual que procesos para su desarrollo.

6.1 REALIDADES DE LOS SISTEMAS EN RED

Las organizaciones modernas son irreversiblemente dependientes de los sistemas de gran escala en red cuya complejidad frecuentemente excede las capacidades ingenieriles actuales en lo que hace al control intelectual. El resultado han sido persistentes dificultades en el desarrollo, gestión y evolución, y fallas intrusiones y compromisos en la operación [Schneider 99]. Estos sistemas se caracterizan por redes heterogéneas de muy gran escala con fronteras y componentes a menudo desconocidos, en los que la conectividad dinámica de sistemas-de-sistemas puede limitar la visibilidad y el control de la seguridad y la supervivencia. Los flujos de tareas de usuarios pueden atravesar sistemas y fronteras con características de seguridad y supervivencia variables.

Además, estos sistemas deben tratar con funcionalidades y calidad de COTS inciertas, comportamientos y vulnerabilidades inesperadas, y una cascada de fallas inter-sistemas imprevistas. La complejidad deriva del comportamiento fuertemente asincrónico del virtualmente desconocido entrelazamiento de comunicaciones entre los componentes del sistema. La Figura 29 describe los componentes de un sistema centrado en red del tipo descripto, el "*Sistema de Combate del Futuro*" (FCS), en el que cada componente es un sistema complejo en sí mismo.

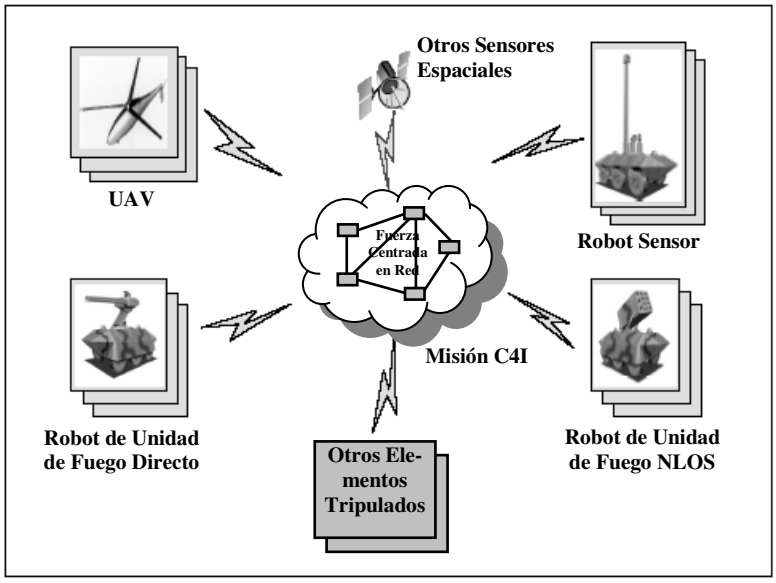


Figura 29. Elementos del "*Sistema de Combate del Futuro*" centrado en red.

El FCS está altamente distribuido, contiene cientos de nodos, está operado por miles de usuarios, conduce comunicaciones y operaciones asincrónicas complejas, está sujeto a sufrir daños, interrupciones y compromiso, y experimenta una continua actualización y evolución. Controlar la complejidad y asegurar la supervivencia son prioridades para el desarrollo de sistemas del tipo de FCS.

Similares características se encuentran en sistemas comerciales centrados en red. Considerando los cruces sistema-a-sistema que incluye una transacción de compra de combustible con una tarjeta de crédito que se describen en la Figura 30, se puede observar que cientos de componentes de hardware y de software son atravesados por múltiples sistemas dentro de las muchas interacciones desde la estación de servicio hacia las comunicaciones vía líneas terrestres y enlaces satelitales, luego hacia las base de datos de la tarjeta de crédito, y de regreso, con muchos resultados posibles. Cada sistema incluido en la red exhibe una funcionalidad y atributos de calidad exclusivos, que incluyen fiabilidad, seguridad y supervivencia.

El agobio de no dominar la complejidad crea dificultades en la comprensión de los sistemas existentes y en la definición de los sistemas que se necesitan. Esto conduce a la pérdida del control intelectual cuando se exceden las capacidades humanas de razonamiento y análisis. ***El control intelectual significa la comprensión del comportamiento del sistema en todos sus niveles y bajos todas las circunstancias de uso***. Esto significa un desarrollo y evolución ordenados, y sin sorpresas en la operación. *El control intelectual no significa la ausencia de incertidumbre, falla o compromisos –ellos son inevitables- sino la previsión y la capacidad de tratar con ellos. Y no requiere de lentos y laboriosos métodos de desarrollo. Por el contrario, el control intelectual permite un rápido desarrollo con confianza. Sin control intelectual, las presunciones y las*

esperanzas están al orden día, y las sorpresas son inevitables[18].

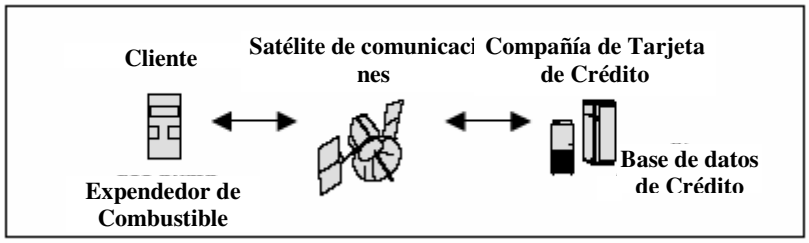


Figura 30. Cruces sistema-a-sistema en la compra de combustible.

Hoy día, se deben hacer frente a temas de complejidad y supervivencia sobre un nuevo nivel de sistemas de gran escala basados en red. La reducción de la complejidad requiere de sólidos principios y métodos de ingeniería para mantener el control intelectual durante la especificación, diseño, y operación. La complejidad y la supervivencia están estrechamente ligadas. *La complejidad disminuye la supervivencia disimulando potenciales fallas y vulnerabilidades, y ocultando caminos de intrusión desapercibidos. La mejora de la supervivencia requiere del conocimiento de las dependencias de los componentes del sistema en todas las situaciones de utilización, del aprestamiento para todos los compromisos y fallas de componentes en todas las situaciones, y del diseño de acciones del sistema para todas las situaciones* [Ellison 99a, Mead 00a]. Resumiendo, ***la supervivencia requiere del control intelectual***.

6.2 INGENIERIA DE FLUJO-SERVICIO-CALIDAD

El desarrollo de sistemas de gran escala basados en red es esencialmente una actividad de integración masiva que busca reconciliar y satisfacer requerimientos de usuario mediante las combinaciones de componentes COTS y a medida, a menudo dentro de una estructura de ambientes predeterminados, sistemas *legacy*, tecnologías de soporte, y dominios de arquitecturas.

En este mundo de sistemas de gran escala, asincrónicos y en red con, a menudo, funcionalidades y estructuras dinámicas e inciertas, se plantean tres preguntas relacionadas con los métodos de ingeniería para la reducción de la complejidad y la mejora de la supervivencia:

1. *¿Cuáles son los principios de ingeniería unificadores para el análisis, especificación, diseño y verificación?*
2. *¿Cómo se deberían especificar y satisfacer los atributos de calidad tales como supervi-*

[18] Las barreras de la complejidad se pueden eliminar mediante principios correctos. Cuando los normandos conquistaron Inglaterra en el Siglo XI, llevaban un censo de qué era lo que habían ganado. Pero nunca fueron calculados los resultados, a pesar del obvio interés de conocer dicha suma. El censo había sido registrado en el sistema romano, y las más brillantes mentes de hoy día fueron doblegadas por la complejidad de sumar muchos números romanos. La representación y los métodos de razonamiento fueron por sí mismos la principal fuente de complejidad. Pero si el censo hubiera sido registrado con notación aritmética decimal, cualquier niño podría haber obtenido las sumas necesarias. Los principios correctos habrían hecho toda la diferencia.

vencia, fiabilidad y desempeño?

3. *¿Cuáles marcos estructurales de arquitectura pueden simplificar el desarrollo y operación del sistema?*

Resumiendo, ¿cuáles son los puntos de anclaje estables y confiables para la especificación y diseño que puede proporcionar una disciplina de ingeniería unificada para el análisis y desarrollo de sistemas de gran escala basados en red?

Los trabajos de investigación están desarrollando nuevas soluciones para responder a estas preguntas. Los siguientes conceptos ayudan a estructurar la línea de investigación:

- **Estructuras de Flujos.** Los flujos de tareas de usuario y sus refinamientos en los usos del servicio del sistema pueden proporcionar principios de ingeniería unificadores para el análisis, especificación, diseño y verificación de los atributos de funcionalidad y calidad.
- **Atributos de Calidad Computables.** Los atributos de calidad pueden estar asociados tanto con flujos como con los servicios del sistema que éstos invoquen, y pueden ser especificados como propiedades funcionales dinámicas que se calculan, en lugar que como predicciones estáticas y a-priori de incierta utilidad en las operaciones de sistemas en tiempo-real.
- **Arquitecturas de Gestión de Flujos.** Las *Estructuras de Flujos* y los *Atributos de Calidad Computables* soportan marcos de referencia de arquitecturas estándares que gestionan flujos, servicios de red, y sus atributos de calidad durante su ejecución.

Las *Estructuras de Flujos* son composiciones de los servicios del sistema llevados adelante por las tareas de usuario para atender las misiones de la organización. Emplean semánticas exclusivas a fin preservar importantes propiedades determinísticas para precisar la comprensión humana y el análisis, a pesar del subyacente comportamiento asincrónico e impredecible de la red. Las *Estructuras de Flujos* toman en cuenta eventos y desenlaces impredecibles que puedan impactar en la supervivencia de la misión.

Los *Atributos de Calidad Computables* de los flujos y servicios que ellos invocan pueden ser gestionados dinámicamente en tiempo de ejecución. Por ello, los conceptos de primer nivel de flujo, servicio y calidad conforman la base de la disciplina emergente de ingeniería de *Flujo-Servicio-Calidad* (FSQ) [Hevner 01, Hevner 02]. Un problema persistente en el desarrollo y gestión de sistemas de gran escala basados en red ha sido la falta de principios de ingeniería unificados independientes de la escala para el control intelectual en la gestión, adquisición, análisis, desarrollo, evolución y operación. La ingeniería de FSQ aborda el problema mediante principios teóricos y métodos prácticos de ingeniería para representar, analizar, desarrollar, y gestionar dinámicamente los flujos del sistema y sus atributos de calidad como artefactos esen-

ciales y principales del desarrollo de los sistemas basados en red.

Los sistemas distribuidos de información son vistos como redes de componentes que se comunican asincrónicamente proveyendo los servicios del sistema cuyas funciones se pueden combinar formando diferentes patrones para satisfacer los requerimientos de la misión de la organización. Los servicios del sistema incluyen todas las capacidades funcionales de un sistema basado en red, desde los protocolos de comunicaciones y los sistemas operativos hasta las bases de datos y las aplicaciones.

El secuenciamiento de los servicios del sistema dentro de los flujos de tareas de los usuarios se puede asociar con los componentes de hardware, software y personal presentes en la red que provee los servicios. Estas composiciones son trazas extremo-a-extremo que definen porciones de las arquitecturas de red cuyo efecto neto es el de llevar adelante las operaciones que satisfacen los requerimientos del usuario. La Figura 31 describe el refinamiento de los flujos de tareas de usuarios en base a los objetivos de la misión desplegados sobre los usos de los componentes de arquitectura del sistema.

Los sistemas de gran escala soportan muchos usuarios simultáneos cumpliendo diferentes roles con muchos posibles flujos de tareas, y servicios del sistema particulares que pueden aparecer una y otra vez en sus definiciones. De hecho, un objetivo principal de diseño en los sistemas de gran escala es la coordinación y la sincronización de múltiples usos de servicios particulares incorporados dentro de los flujos. En las redes dinámicas con constantes variaciones de la función y el uso, los flujos y sus correspondientes trazas en la arquitectura de los servicios del sistema [Parnas 94] actúan como principios estables para la especificación y diseño funcional y no-funcional, esto es, atributos de calidad. Durante la ejecución, los servicios invocados por los flujos pueden experimentar un aluvión de entrelazamientos de usos asincrónicos que desafían a la comprensión humana.

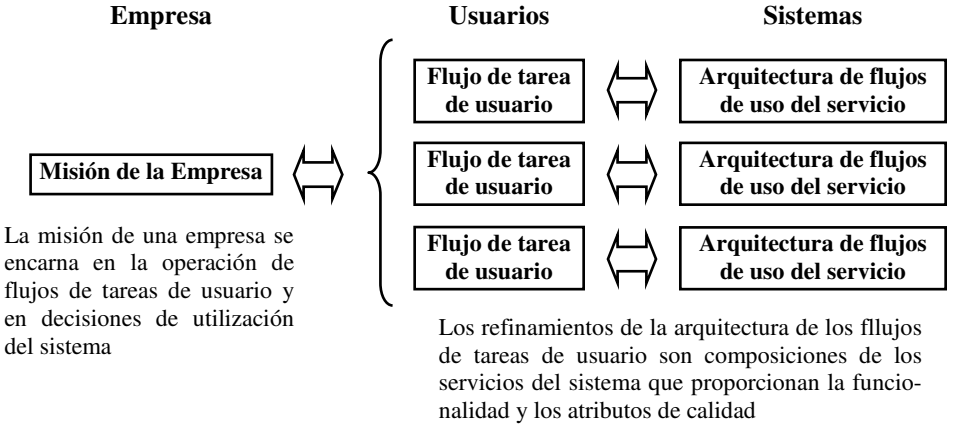


Figura 31. Refinamiento de flujos de tareas de usuario en usos de servicio del sistema.

Una propiedad clave de las *Estructuras de Flujos* es un principio semántico que permite que los flujos exhiban propiedades determinísticas para una comprensión humana y un análisis más directo a pesar del comportamiento asincrónico subyacente. En consecuencia, *los flujos se representan como simples estructuras procedurales conformadas por invocaciones y computaciones locales anidadas y secuenciadas, expresadas en términos de estructuras ordinarias de secuencia, alternancia, iteración y concurrencia; dichas estructuras definen un álgebra de composición de componentes con propiedades deseables*.

Por ejemplo, las *Estructuras de Flujos* preservan la efectividad de los métodos de razonamiento y el refinamiento, abstracción y verificación del refinamiento referencial para la comprensión humana. Los flujos se pueden expresar en virtualmente cualquier lenguaje que utilice primitivas de *Estructuras de Flujos* para especificar las tareas de usuario que hacen uso de los servicios del sistema en términos precisos. Los servicios invocados por los flujos se pueden refinar en flujos que invocan a otros servicios, etc. , en un proceso de diseño recursivo que emplea estructuras y métodos de ingeniería idénticos en todos los niveles de refinamiento. Las *Estructuras de Flujos* están relacionadas superficialmente con los métodos *workflow* [Leymann 00], pero definen rigurosos principios independientes de la escala para el análisis, desarrollo y operación de sistemas de gran escala.

Los flujos se pueden organizar en *Conjunto de Flujos* (*FlowSets*) relacionados, asociados con componentes y particiones de red particulares. El análisis transitivo de los flujos puede revelar dependencias a menudo desapercibidas. Los flujos definen los niveles de atributos de calidad requeridos ellos mismos, como así también para la ejecución de los servicios que ellos referencian. En la Figura 32 se describen las operaciones de ingeniería FSQ para sistemas existentes y nuevos.

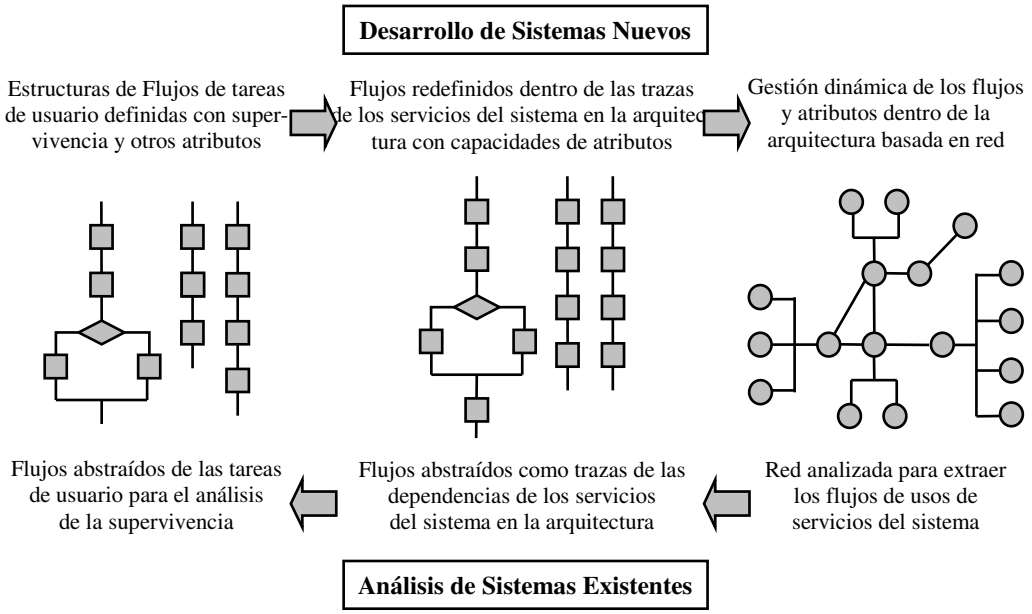


Figura 32. Operaciones de ingeniería FSQ para sistemas nuevos y existentes.

En el caso de los sistemas nuevos, la especificación de flujo comienza con las tareas de usuario que soportan los los objetivos de la misión de la organización, asegurando de este modo una solución centrada en el usuario para el diseño y el desarrollo. En particular, ***los flujos son vehículos para la definición de los atributos de calidad requeridos, tales como fiabilidad y supervivencia***. Algunos flujos requieren niveles superiores de fiabilidad y supervivencia que otros, y la definición de de los requerimientos de atributos específicos de un flujo permite establecer un equilibrio entre los costos y beneficios informados en el diseño del sistema.

En el caso de los sistemas existentes, los flujos de las operaciones de misión crítica se pueden extraer y analizar a fin de revelar las dependencias desapercibidas y los puntos únicos de falla. Este análisis permite la identificación y desarrollo de mejoras en la supervivencia.

Resulta importante denotar que ***los flujos se pueden definir tanto para usos legítimos como ilegítimos***. La utilización del sistema del intruso se puede expresar en flujos que revelen los componentes pasibles de compromiso y ayuda a definir las mejoras en la seguridad y la supervivencia [Mead 00a, Moore 01a].

En términos de una comprensión intuitiva, ***los flujos se pueden pensar como jerarquías determinísticas de usos de servicios superpuestos con otros flujos para su ejecución asincrónica sobre una red a gran escala***. Este concepto se describe en la Figura 33 para el caso de un flujo del ejemplo del "*Sistema de Combate del Futuro*". Los flujos definen un uso estructurado de las capacidades de la red mediante la definición de las comunicaciones entre servicios del sistema basado en red, y de las composiciones de dichos servicios.

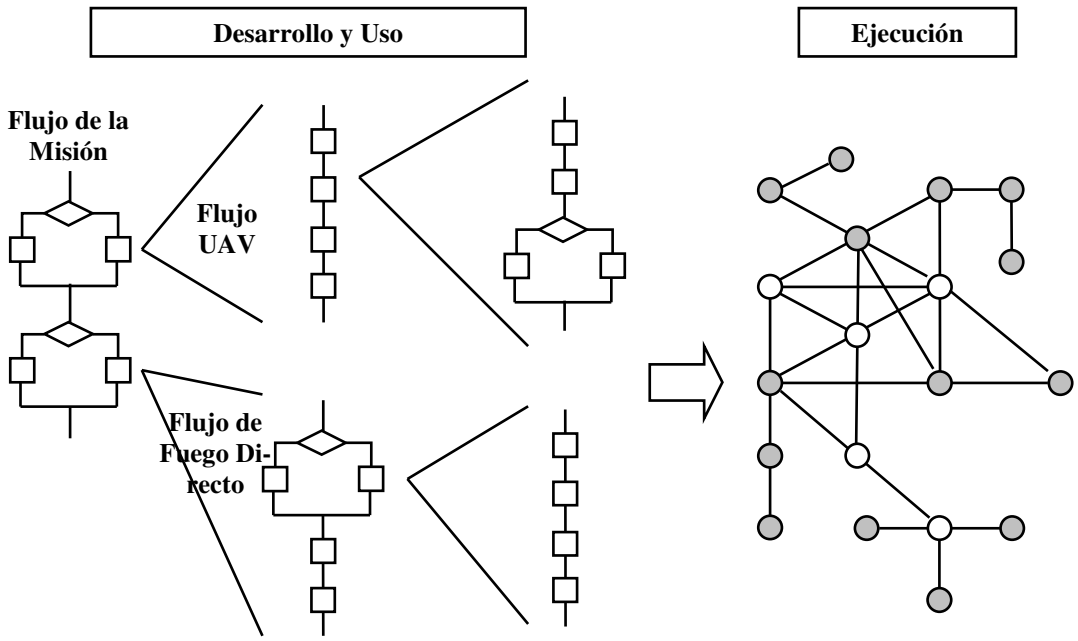


Figura 33. Superposición de un flujo determinístico sobre una red asincrónica.

En la ingeniería FSQ, *atributos de calidad tales como disponibilidad, fiabilidad, y supervivencia se definen como funciones computacionales, y están asociadas tanto con flujos como son servicios*. ***En el pasado se ha destinado sustancial esfuerzo al desarrollo de caracterizaciones a-priori de los atributos de calidad***. *En lugar de focalizar la atención sobre métodos descriptivos de valor limitado para las redes dinámicas, se ha adoptado una solución alternativa y se pregunta de qué manera se pueden definir, computar y poner en práctica tales características de la operación del sistema*. ***Esto es, se desea definir los atributos de calidad como funciones ha ser computadas, más que como descripciones estáticas de las capacidades ha ser alcanzadas***.

En tanto tales funciones dependen de cómo han de ser computadas y, por ende, pueden diferir de las visiones tradicionales de los atributos de calidad, éstas pueden posibilitar nuevas soluciones al análisis, diseño, métricas y evaluación operacional de atributos. Un aspecto clave de la solución computacional es *la habilidad de asociar atributos de calidad con flujos específicos en lugar de hacerlo con sistemas completos*, permitiendo, en consecuencia, la diferenciación entre las capacidades de atributos en base a la criticidad de la misión en el dominio de la ingeniería de la supervivencia. Algunos atributos de calidad, tales como disponibilidad y fiabilidad son fácilmente computados en el análisis computacional. Otros, tales como seguridad y supervivencia serán más difíciles. No obstante, se piensa que el esfuerzo debe ser hecho, y que están a la vista nuevas soluciones y perspectivas.

En un mundo de las *Estructuras de Flujos* y los *Atributos de Calidad Computables*, resulta natural considerar marcos estructurales de la arquitectura del sistema basados en la gestión de flujos dinámicos y de atributos de calidad [Sikora 98, Haekel 99, Sullivan 99]. *Una tarea fundamental de control en los sistemas de gran escala es la gestión del secuenciamiento de los servicios del sistema para satisfacer las especificaciones de flujo. Los conceptos FSQ sugieren marcos estructurales independientes del tema denominados* ***Arquitectura de Gestión de Flujo FMA*** (*Flow Management Architecture*). Estos marcos estructurales podrían compatibilizar requerimientos de flujo con disponibilidades de servicios, e implementar estrategias de gestión operacional basadas en una red dinámica y en las capacidades de servicios, y en cargas de trabajo.

Las FMAs encarnan el concepto de un ***Gestor FSQ*** *(FSQ Manager), centralizado o descentralizado dentro de la arquitectura de un sistema, el cual provee la gestión de flujos. En particular, el Gestor FSQ proporciona una evaluación de los atributos de calidad dinámicos*. Por ejemplo, la gestión de la supervivencia incluye una variedad de estrategias tales como caminos de comunicaciones alternativos, sustitución de recursos, estado de reconstrucción, aprovisionamiento alternativo, y re-inicialización y re-configuración del sistema. Se deberá diseñar e instanciar un *Gestor FSQ* en una variedad de formatos y tecnologías, dependiendo de los requerimientos de usuario, la configuración de la red y del ambiente operacional.

La ingeniería FSQ puede reducir la complejidad y adicionar claridad al desarrollo de sistemas basados en red. Se pueden diseñar y verificar las especificaciones de las Estructuras de Flujos asociadas con las tareas de una organización con una completa compresión humana, con varios niveles de abstracción, en un proceso integrado que va desde los flujos de tareas del usuario hasta los componentes arquitectónicos. Las Estructuras de Flujos prescriben las conexiones y operaciones de red lógicas, definen la capacidad de composición entre nodos y servicios, y soportan tanto control centralizado como distribuido. La especificación del comportamiento del sistema basado en red y la conectividad lógica se define mediante el Conjunto de Flujos de sus usos de servicio. La especificación de cada servicio dentro de un sistema en red incorpora todos sus usos dentro de todos los flujos en los cuales éste aparece. Los marcos estructurales FMAs proveen plantillas sistemáticas para la gestión de los instanciaciones de los flujos y la compatibilización de los Atributos de Calidad Computables.

6.3 SEMANTICAS DE LA ESTRUCTURA DE FLUJOS

En los sistemas basados en red de gran escala, los flujos pueden abarcar una importante cantidad de nodos de red y de enlaces de comunicación, en los que el comportamiento de los servicios invocados no siempre se puede conocer y predecir. En este ambiente, se deben gestionar una variedad de ***Factores Inciertos*** (*Uncertainty Factors*), que incluyen:

- **Función imprevisible**. Un servicio puede ser provisto por componentes COTS o por un Proveedor de Servicio Externo de funcionamiento y disponibilidad imprevisibles que pueden no realizar las operaciones esperadas o en el momento en que son invocadas.
- **Función comprometida**. Un servicio puede haber sido comprometido o interrumpido por una intrusión o un ataque físico y puede no ser capaz de realizar su función o hacerlo de manera correcta.
- **Función de alto riesgo**.Un servicio puede no proveer los adecuados niveles de atributos de calidad (QoS) requeridos por el flujo.
- **Función asincrónica**. Un servicio puede ser usado simultáneamente y en forma asincrónica por otros flujos y, en consecuencia, producir resultados dependientes de una impredecible historia de uso, tanto legítimo como ilegítimo.

Estos factores son realidades siempre presentes en el comportamiento de los sistemas basados en red de gran escala [Schneider 99]. Tratar con ellos es un problema de gestión del riesgo de la organización con potenciales consecuencias serias. Resulta importante detectar cuándo han ocurrido y tomar las acciones apropiadas para continuar operando dentro de los ambientes que ellos han creado. Para el caso de los flujos de misión crítica, estas acciones deben asegurar la super-

vivencia, sin importar cuáles sean los ambientes que se presenten [Mead 00a].

En el mundo actual, es imprudente desde una perspectiva de la gestión del riesgo fallar en la atención de los *Factores Inciertos* en todos los niveles de operación de la organización y del sistema.

Se han definido semánticas matemáticas de las *Estructuras de Flujos* de soporte para el desarrollo y la verificación de flujos para estos ambientes inciertos como una práctica estándar de ingeniería. *Para permitir un comportamiento imprevisible de los servicios, las semánticas de flujos permiten especificar sólo el procesamiento que realiza un flujo por sí mismo, y no el procesamiento de los servicios que él invoca.* La ingeniería de *Estructuras de Flujos* requiere de la definición de las acciones apropiadas mediante aquellos flujos de todas las posibles respuestas de los servicios claves, tanto deseados como indeseados. *En consecuencia, si el comportamiento de los servicios invocados cambia por cualquier razón, la especificación y verificación del flujo invocante no necesita cambiar*. Esta solución da cabida a las realidades de los actuales sistemas basados en red y ofrece importantes ventajas. Para la supervivencia de la misión, requiere que los *Factores Inciertos* sean tratados de manera explícita durante el diseño, y de esta manera, se contemplan importantes aspectos de la gestión del riesgo de la organización. Esto permite que los flujos y el razonamiento asociados a ellos se encuentren localizados y completos. Y permite definir los flujos mediante simples estructuras determinísticas a pesar del comportamiento asincrónico subyacente de los servicios que los constituyen. Estas estructuras determinísticas se pueden refinar, abstraer, y verificar mediante métodos de composición directos para la comprensión y el control intelectual humanos.

Resulta evidente que estos objetivos requieren de la extensión del modelo tradicional de las semánticas funcionales. *El modelo semántico FSQ está basado en el concepto bien conocido de servicios como reglas para las funciones matemáticas (o relaciones si los flujos incluyen operaciones concurrentes), esto es, asociaciones desde dominios (entradas, estímulos) hacia rango (salidas, respuestas)* [Linger 79, Mills 86, Prowell 99, Hoffman 01, Mills 02]. La extensión clave requerida para encarar sistemáticamente los *Factores Inciertos* es la construcción de las historias de las invocaciones de servicios como parte de la conducta especificada de los flujos. Matemáticamente, esto se logra mediante la inclusión de la ***Historia de Invocación de Estímulos ISH*** (*Invocation Stimulus History*) de cada servicio dentro del rango de la función que representa la especificación de un flujo. Además, debido a que el subsiguiente procesamiento del flujo puede depender de las respuestas a estas invocaciones, la ***Historia de Respuestas de Invocaciones IRH*** (*Invocation Response History*) debe ser parte del dominio de la función matemática que representa la especificación de un flujo. El diagrama de la Figura 34 ilustra estas semánticas para un flujo F que invoca a un servicio A.

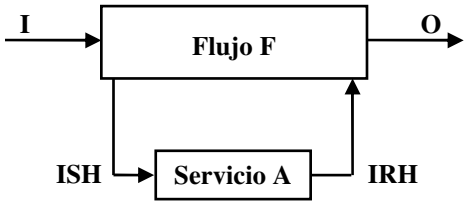


Figura 34. Elementos de las semánticas de Flujo-Servicio

I es e conjunto de posibles entradas del flujo F, y O es el conjunto de posibles salidas desde el flujo F. En consecuencia, las semánticas de F pueden estar expresadas mediante una función matemática **f** dentro del dominio **IxIRH** y el rango **OxISH**. Es esta inclusión nada intuitiva de las respuestas del servicio dentro del dominio de F y del estímulo del servicio dentro del rango de F lo que permite que los flujos manejen los Factores Inciertos. En particular, IRH representa el rango de las respuestas posibles del servicio y, en consecuencia, representa las posibilidades del *Factor Incierto* que tendría que tenerse en cuenta en el diseño del flujo. Hacer frente a los *Factores Inciertos* requiere de valorar y poner en práctica todas las respuestas posibles, deseadas e indeseadas, que pueden producir las invocaciones de servicio. Por supuesto, ninguna semántica puede forzar tal diseño, sino que sólo puede iluminar lo que resulta deseable de hacer.

En este modelo semántico, no se requiere la especificación del flujo F para representar los comportamientos que resultan debido a la invocación del servicio; simplemente define la invocación del servicio A con ciertos parámetros, y de qué manera la respuesta a esa invocación afecta el subsiguiente procesamiento de F. Esto significa, por ejemplo, que cualquier servicio de bajo nivel invocado por el servicio A no necesita ser parte del ISH y del IRH del flujo F. Si así fuera, la especificación de F no debería cambiar si el servicio A fuera modificado, por ejemplo, para invocar diferentes servicios de bajo nivel. Esta solución difiere de las semánticas funcionales tradicionales, en las que la especificación de F debería requerir la inclusión de todos los efectos sobre todas las invocaciones de servicios de bajo nivel por parte del servicio A como una parte de su especificación funcional.

Esta solución para la especificación es clave para el mantenimiento del control intelectual sobre la especificación y diseño del flujo. Como se denotara, se pueden modelar flujos determinísticos que invocan servicios no-determinísticos asincrónicos mediante funciones matemáticas determinísticas, haciendo mucho más simple el razonamiento y el análisis humano. Alternativamente, si el comportamiento de los flujos fuera no-determinístico, entonces los mismos flujos se podrían volver mucho más complicados, y podrían necesitar que sus semánticas fueran expresadas como una relación matemática desde el dominio IxIRH hacia el rango OxISH. Esta situación compleja se evita mediante las semánticas FSQ.

Las semánticas de flujos descriptas anteriormente se adecúan de manera particular con la situación

común en la que el servicio A ya existe sobre una red, o es provisto por componentes COTS o a medida con funciones complejas y, posiblemente, desconocidas. En casos en los que el servicio A es nuevo y debe ser diseñado como parte de la implementación del flujo F, estas semánticas de flujos se pueden combinar con métodos tradicionales de diseño y verificación tal como aquéllas que se encuentran en las estructuras basadas en objetos [Mills 86] para soportar el razonamiento acerca del comportamiento combinado del sistemas compuesto por F y A. De esta manera, el comportamiento deseado de F y de A se puede utilizar para guiar la construcción de A. En particular, las estructuras proveen representaciones orientadas a la historia, el estado y el procedimiento de flujos y servicios, y métodos para su abstracción, refinamiento y verificación.

Los conceptos y técnicas de las *Estructuras de Flujos* aplicados a los flujos de red se pueden escribir en casi cualquier lenguaje imperativo, incluidos C++ y Java, siempre que el lenguaje incluya el conjunto básico de estructuras de control, en particular, secuencia, alternancia e iteración. Las especializaciones y extensiones de estas estructuras también resultan valiosas. Las invocaciones de servicio en estos lenguajes son llamados a métodos sobre objetos. Se puede definir un ***Lenguaje de Estructura de Flujo FSL*** (*Flow Structure Lenguaje*) que captura la esencia de los conceptos FSQ independientemente de la sintaxis del lenguaje de implementación específico. Para hacer frente a la concurrencia dentro del propio flujo, también se ha incluido en FSL una estructura concurrente. Debido a que los aspectos específicos de los tipos de datos y la sintaxis de declaración del lenguaje no afectan la aplicabilidad de FSQ, estas características pueden perder énfasis en el FSL. La Figura 35 ilustra las típicas estructuras de control de FSL.

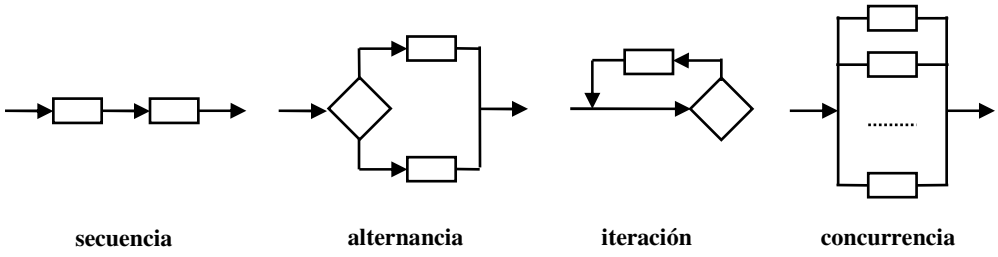


Figura 35. Estructuras de control típicas de FSL.

El comportamiento global de un flujo particular es el siguiente: se invoca un flujo con valores asignados a sus parámetros de entrada, y al finalizar la ejecución del flujo, los valores finales de salida de los parámetros se devuelven al invocador del flujo, ya sea un usuario humano u otro flujo. Un flujo puede definir datos locales no persistentes para almacenar valor intermedios producidos por las computaciones del flujo. Finalmente, un flujo puede invocar servicios para realizar varias actividades en red o locales, incluidas almacenado, acceso o modificación de datos persistentes. El diseño del flujo implica que el estado persistente requerido por un flujo debería estar encapsulado dentro de los servicios.

Además de las estructuras de secuencia, alternancia, iteración y concurrencia, el FSL contiene una sentencia "*use*" para invocar servicios. La sentencia invoca predicados post-fix para evaluar y poner en práctica equivalencias de clases definidas por el diseñador sobre el conjunto de todas las posibles respuestas, tanto deseado como no-deseadas. Esta división y análisis de equivalencia de clases de la respuesta atiende el requerimiento de hacer frente a los *Factores Inciertos* característicos del comportamiento en red, y a las implicancias sobre la supervivencia que ellos imponen. Los diseñadores de flujos seleccionan los servicios claves para este análisis de respuesta. La sintaxis general de la sentencia *use* es la siguiente:

```
use <service>.<method>(<parameters>)
  response <status_variable> is
       <enumerated_value_1> when <expression>
     | <enumerated_value_2> when <expression>
     | ...
     | <enumerated_value_n> when <expression>;
```

Por ejemplo, la siguiente sentencia *use* ilustra la invocación a una base de datos de reservas de avión para realizar una reserva en un vuelo:

```
use Airline.reserve(customer, flight, date, result, seat)
    response status is
         NOTRESERVED when result = false
       | RESERVEDNOSEAT when (result = true) and (seat = "")
       | RESERVEDWITHSEAT when (result = true) and (seat !="");
```

Además de la enumeración explícita de las equivalencias de clases sobre la respuesta, también se pueden evaluar los parámetros basados en la red y el estado del componente (por ejemplo, NOTCONNECTED, NORESPONSE). Tales evaluaciones son importantes en cuanto a la valoración y puesta en práctica de las propiedades dinámicas de una red.

Un conjunto de ***Teoremas FSQ*** captan y exploran los fundamentos de las semánticas FSQ. En el Anexo 5 se describen teoremas de ejemplo. Las pruebas están más allá del alcance de este trabajo, y se pueden encontrar en [Pleszkoch 02].

6.4 OPERACIONES DE INGENIERIA DE ESTRUCTURA DE FLUJOS

Las *Estructuras de Flujos* soportan muchas operaciones de ingeniería en el análisis y desarrollo de sistemas basados en red. A continuación se describe brevemente una operación representativa, y en el Anexo 5 se presentan otras más.

Ingeniería de Flujos para Factores Inciertos

La ingeniería de *Factores Inciertos* requiere que los flujos abarquen a todas las posibles respuestas (IRH) provenientes de los servicios críticos. Este proceso de ingeniería requiere de la definición de

predicados *post-fix* sobre las respuestas para determinar y diseñar acciones apropiadas.

Los diseñadores pueden organizar las respuestas de acuerdo a las equivalencias de clases dependientes del tema de interés. Estas equivalencias de clases son el tema de la gestión de riesgo y de la continuidad de la misión en la ingeniería de supervivencia. Les corresponde a los diseñadores seleccionar las invocaciones a los servicios críticos a los que debería someterse el análisis de la respuesta.

La Figura 36 ilustra el uso de predicados *post-fix* en un fragmento de un flujo de control de tráfico aéreo ficticio. Un controlador que utiliza el flujo está a la espera de identificar una aeronave (*use a/c ident*) y obtener su posición (*use a/c position fix*). Tres predicados *post-fix* siguen a la invocación del servicio *a/c ident*. Estos predicados realizan el *parsing* en equivalencia de clases de acuerdo a que si fue recibida alguna respuesta, si la misma fue una identificación de aeronave, y si la misma fue una identificación válida.

En este pequeño ejemplo, cada caso de evaluación negativa se le notifica al controlador mediante la interfase de servicio del controlador. Pero se debe prestar atención a los problemas al momento del diseño y las discusiones durante el desarrollo de un sistema de soporte de este flujo de misión crítica. La falla del servicio *a/c ident* es un problema muy serio que impacta en la realización del flujo y, en consecuencia, en la seguridad de la aeronave. El análisis transitivo de otros flujos de los cuales depende *a/c ident* puede revelar una serie de posibilidades de fallas en cascada a la que se debe atender para garantizar la supervivencia de este servicio crítico. Este análisis puede dar por resultado cambios importantes en la arquitectura propuesta del sistema. En cada caso, debería quedar en claro que la notificación al controlador es una acción insuficiente para este problema serio, y que este flujo debe ser rediseñado.

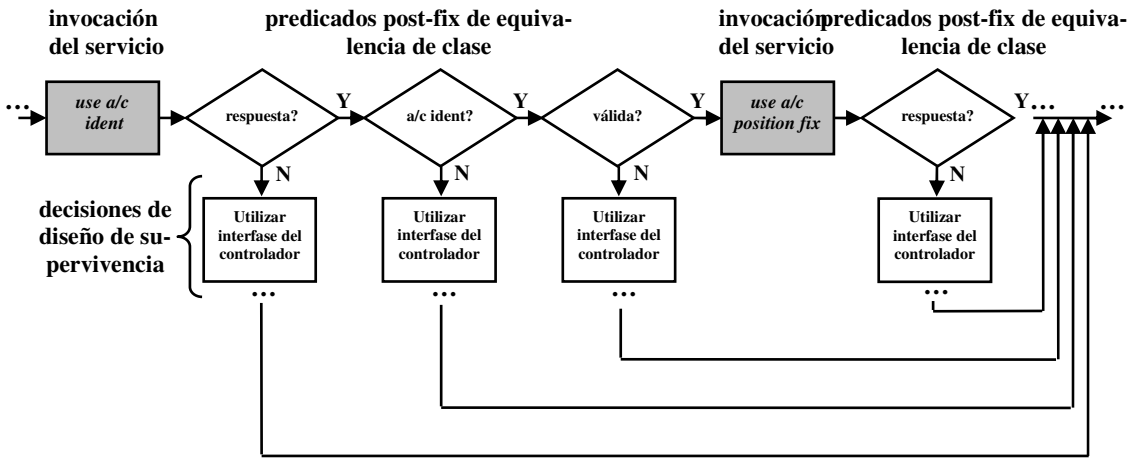


Figura 36. Evaluación de respuesta de servicio *post-fix* para el análisis de la supervivencia y la gestión de riesgo.

Debe observarse en esta discusión que las semánticas de las *Estructuras de Flujos* le permiten a

los diseñadores definir y realizar la equivalencia de clases de respuesta que incluyan los *Factores Inciertos*. De esta manera se respalda la gestión de riesgo de la organización, la cual analiza todas las posibles repercusiones, y gestiona la supervivencia, lo que requiere de acciones para todas las posibles repercusiones.

6.5 ATRIBUTOS DE CALIDAD COMPUTALBES

A menos que un sistema esté implantado sobre una red interna cerrada, muchos detalles del sistema pueden ser desconocidos. No obstante, las organizaciones ponen demandas extraordinarias sobre los sistemas en lo que hace a fiabilidad, disponibilidad, seguridad, y otros atributos de calidad claves [Haeckel 99]. Se ha venido haciendo un sustancial esfuerzo en el desarrollo de caracterizaciones descriptivas, y a veces subjetivas, de los atributos de calidad, como por ejemplo, atributos de supervivencia [Ellison 99a, Sullivan 99].

En lugar de centrar el trabajo en los métodos descriptivos, la solución ***Atributos de Calidad Computables CQA*** (*Computacional Quality Attributes*) define, computa y actúa sobre los atributos de calidad como características dinámicas de la operación del sistema. A continuación se describen los principios matemáticos y los marcos estructurales para estas operaciones.

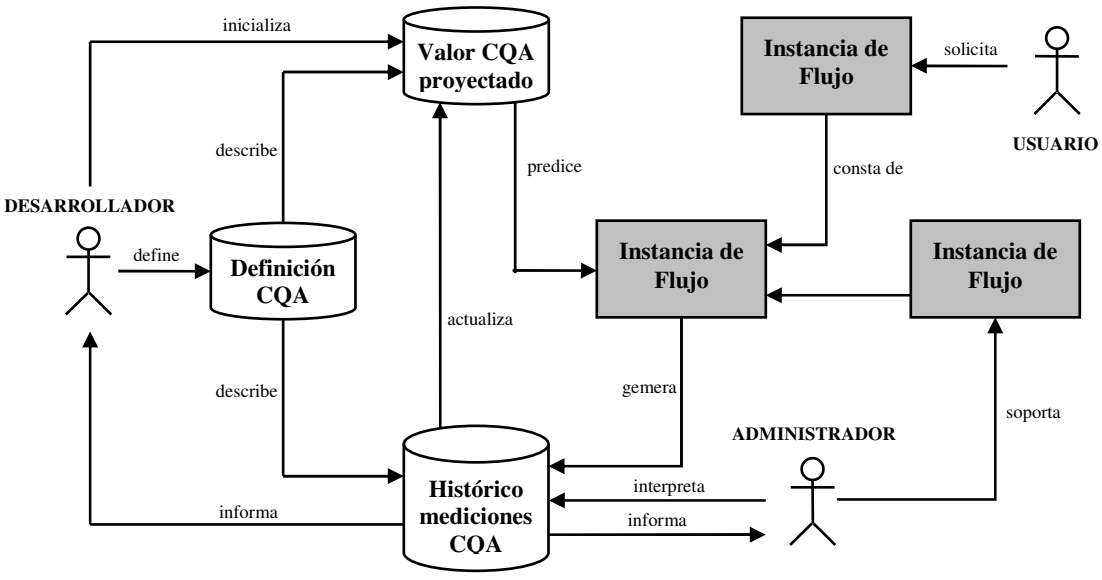


Figura 37. La solución Atributos de Calidad Computables.

Muchos investigadores han establecido atributos de calidad basados en componente, tales como fiabilidad, desde la perspectiva del sistema como un todo [Siegrist 88, Krishnamurthy 97, Gokhale 98, Yacoub 99, Hamlet 01]. *Sin embargo, desde la perspectiva de un usuario de un sistema distribuido, no existe la necesidad de tener una visión de los atributos a nivel del sistema. El usuario está interesado en la provisión de los servicios esenciales, no del estado del*

sistema. ***La solución CQA atiende al interés del usuario****.*

Los atributos de calidad se definen en el nivel del servicio, compuesto con los niveles de servicio y de flujos, y evaluado en el nivel del servicio o del flujo, o en ambos niveles, dependiendo de la solicitud CQA especificada por el usuario. Esta solución CQA, que se ilustra en la Figura 37, proporciona una marco semántico consistente para la adquisición de información del desempeño de los atributos de calidad para los flujos y servicios esenciales, y para el cómputo de las capacidades de calidad del sistema relacionadas con la gestión de la ejecución del flujo.

Los CQAs están definidos como una asociación funcional de utilización (es decir, el dominio de entrada para un uso, ambiente y tiempo particular) con valores de atributo que representan una medida de calidad. *Este método soporta la descripción de cualquier conjunto de atributos de calidad y de cualquier modelo utilizado para describir cada atributo, proporcionando a cada modelo un valor numérico.*El *desarrollador* define los CQAs de interés e inicializa los valores CQA proyectados para cada servicio en base a las afirmaciones de los fabricantes, estimaciones, registros históricos, etc. Mientras el sistema se encuentra operando, se realizan en forma iterativa las siguientes actividades a medida que los servicios son ejecutados:

- Se monitorean las ejecuciones de los servicios y se miden sus atributos de calidad.
- Se acumulan los valores de atributos de calidad.
- Se analiza el histórico acumulado y se lo utiliza para actualizar los valores CQA proyectados como apropiados (Tener en cuenta que una observación única prácticamente casi no provee información; sin embargo, a lo largo del tiempo, el peso acumulativo de las historias de ejecución permiten saber los métodos para predecir los CQAs para las instancias de servicio).

El *usuario* solicita los flujos de servicios que han de ser realizados por el sistema. Las solicitudes de flujos incluyen los niveles requeridos de CQAs correspondientes, como así también para la ejecución de los servicios particulares que ellos referencian. En consecuencia, los CQAs solicitados por el usuario definen las restricciones para la ejecución del flujo.

El *administrador* de sistema utiliza el histórico de CQA como fundamento para la toma de decisiones que tienen que ver con las actualizaciones de los sistemas distribuidos (por ejemplo, balance de carga y replicación de servicios). El conocimiento que posea el administrador de sistema sobre el estado del sistema y sus servicios puede proveer una valiosa percepción en la interpretación del histórico de CQA.

Debido a que los CQAs de cada servicio se pueden calcular en forma independiente de los de otros servicios y componentes del sistema, tanto la implementación del componente como la distribución de las instancias del componente pueden proporcionar una variedad de oportuni-

dades para conseguir los valores deseados para los CQAs de interés. Durante la implementación de un servicio, el diseñador de un componente puede contar con múltiples opciones, cada una con diferentes valores de atributos de calidad. A su vez, el administrador de sistema puede distribuir una o más instancias de una implementación particular de un servicio a lo largo de la red, haciendo que cada distribución pueda presentar valores diferentes de los CQAs.

6.5.1 Definición CQA

El modelo general para la definición de un CQA, q_i, es una jerarquía de definiciones funcionales:

$$qi = f(q_{i,1}, q_{i,2}, ...), \text{ donde } q_{i,1} = g(q_{i,1,1}, q_{i,1,2},...)$$

Cada CQA posee un significado preciso y una representación funcional formal.

En general, el ***proceso de definición CQA*** se puede describir de la siguiente manera:

- Determinar los atributos de calidad de interés.
- Definir en forma recursiva las funciones de un atributo de calidad como una jerarquía de definiciones funcionales de atributos hasta que cada atributo "hoja" haya sido definido como una función computacional.
- Determinar la estructura de datos y los procedimientos para el almacenado de los valores de atributo.

Observar que la cuantificación y el almacenado de los valores CQA en niveles detallados pueden impactar de manera adversa sobre el desempeño del sistema; en consecuencia, generalmente se requiere de un análisis para fin de determinar la apropiada granularidad de los valores que han de ser almacenados. Los problemas generalmente incluyen la frecuencia de cálculo, y si almacenar los valores reales o promedios.

La solución para la definición CQA permite contemplar todos los grados de complejidad en la definición de atributo de calidad. Cualquier número de atributos pueden ser de interés para los usuarios de un sistema distribuido, incluidos atributos que describen los requerimientos de comportamiento y los atributos que describen las características de desempeño. Algunos atributos (tales como disponibilidad, fiabilidad y tiempo de respuesta) cuentan con una rica literatura y se los puede definir fácilmente como CQAs. Otros atributos han de requerir mayor esfuerzo para desarrollar una definición de CQA; por ejemplo, la supervivencia se puede definir como una función de la arquitectura del sistema y de otros atributos, a su vez, la arquitectura del sistema se puede definir como una función de una variedad de atributos que incluyen la conectividad, etc. Algunos ejemplos de definiciones de CQAs se proveen en [Walton 02].

6.5.2 Análisis de solicitud de Flujo

A fin de implementar el marco de trabajo CQA, los servicios de un sistema basado en red deben estar "attribute-enabled" para los CQAs que son de interés para los usuarios. Esto significa que ***debe existir un mecanismo de soporte para la evaluación y reporte CQA***. *Este mecanismo debe estar implementado directamente a nivel del servicio, o computado mediante la evaluación y composición de CQAs correspondientes a servicios de menor nivel*. Un CQA para el cual un servicio no está "*attribute-enabled*" poseerá un valor de "0" en caso que alguna solicitud de servicio incluya una restricción sobre ese atributo.

Los CQAs de menor nivel se componen a fin de establecer el valor de los CQAs para el siguiente nivel superior. Los CQAs también se componen entre servicios para establecer las propiedades del flujo. Para garantizar la simplicidad, la solución CQA asume que las ejecuciones del servicio son independientes (para asumir algo diferente se debería tener conocimiento de los diseños e implementaciones de cada uno de los componentes que constituyen cada servicio). *Debido a que los CQAs asocian utilización con valores de atributo, resulta importante que cada valor CQA se determine en base al dominio de interés de utilización del flujo*.

Dada una solicitud de flujo restringida y un conjunto de flujos candidatos, el objetivo del análisis de solicitud de flujo es la determinación de si cada flujo candidato satisface el conjunto de restricciones del CQA. Si cada restricción de CQA está definida como un valor mínimo aceptable, entonces el conjunto de flujos aceptables se encuentra dentro de una región convexa de calidad aceptable como fuera definido por las restricciones CQA[19]. A continuación se ofrece una descripción de alto nivel y un ejemplo simple para ilustrar los conceptos.

Un usuario especifica una solicitud de flujo de la siguiente manera:

1. Se define el flujo como una composición de servicios del sistema para realizar las tareas de usuario.
2. Se determinan los CQAs de interés para esta solicitud de flujo en base a los dominios correspondientes a este uso, ambiente y tiempo particulares.
3. Se especifican las restricciones para los servicios solicitados y las restricciones para el flujo en términos de valores aceptables o rangos de valores para los CQAs.

Una solicitud de flujo se procesa de la siguiente manera:

1. Se identifican los flujos candidatos para proporcionar la composición solicitada de los servicios del sistema.
2. Cada flujo candidato se evalúa en base a si el conjunto proyectado de valores CQA asociados con los servicios incluidos en él satisfacen las restricciones de CQA de la solici-

[19] En Walton 02 se proporcionan detalles sobre la composición CQA y análisis de solicitud de flujo.

tud de flujo. (Si múltiples flujos candidatos satisfacen la solicitud de flujo, se puede realizar una negociación con el usuario para seleccionar el flujo óptimo en base a prioridades especificadas por el usuario)

Los flujos candidatos se evalúan de la siguiente manera:

1. Además de los servicios solicitados, se representan todas las conexiones de red como servicios.
2. Se predicen los valores para cada CQA de interés para cada servicio dentro del flujo. (Si un servicio no está "*attribute-enabled*" para un CQA de interés, se le asigna un valor "0" para ese CQA.)
3. Se chequean las restricciones CQA especificadas para cada servicio mediante la comparación de los valores CQA pronosticados con las restricciones. Si se satisfacen todas las restricciones a nivel del servicio, se continúa con el siguiente paso; de lo contrario, se devuelve una respuesta "NO" a la solicitud.
4. Se chequean las restricciones CQA a nivel del flujo:
    a. Se desarrolla un modelo gráfico basado en el estado que incluya todas las posibilidades de satisfacción de las restricciones CQA, incluidos los resultados adversos.
    b. Se convierte cada CQA en una distribución de probabilidad y se etiquetan los arcos del modelo gráfico con la distribución de probabilidad, produciendo un modelo probabilístico. (Se utilizan los pesos en la medida que las situaciones planteen unidades CQA que no son uniformes para todos los servicios.)
    c. Se computa el CQA a nivel de flujo proyectado en base al modelo probabilístico.
5. Si el CQA a nivel de flujo satisface las restricciones a nivel de flujo, se devuelve un "SI"; de lo contrario, se devuelve un "NO".

6.5.3 Consideraciones para el Análisis CQA

Las fallas transitorias y las consideraciones sobre cambios dinámicos en los CQAs a lo largo del dominio de entrada pueden hacer que la composición resulte problemática; por ello, se requiere del uso de políticas y suposiciones tales como las siguientes:

- Si un recurso no está disponible cuando una instancia de flujo lo necesita, la instancia de flujo abortará sin quedar a la espera del recurso.
- Los flujos no pueden reservar recursos.
- La ejecución del flujo no incluye anticipaciones.
- No se realizan reparaciones ni reemplazos durante un flujo.
- Las fallas de componente y de servicio son estadísticamente independientes.

- Las distribuciones de probabilidad asociadas con los CQAs no cambian durante la ejecución de un flujo.

Se requiere haber alcanzado un cierto grado de comprensión tanto del uso como de la replicación antes de realizar la composición de los CQAs. Por ejemplo, si se tienen tres nodos que ejecutan el mismo servicio con los mismos datos, la falla de un nodo no es problema a menos que los tres nodos replicados fallen; en cambio, si los tres nodos ejecutan el mismo servicio con datos diferentes, la falla de un nodo particular provoca una falla en el flujo que ha requerido la ejecución del servicio sobre los datos de ese nodo.

6.5.4 Actualizaciones dinámicas de los Atributos de Calidad Computables

El algoritmo para la actualización de los CQAs proyectados a partir de la información histórica debe ser sensible al hecho de que el sistema se encuentra continuamente evolucionando.

El reemplazo o mantenimiento de un nodo, de un enlace de comunicaciones o de un servicio puede cambiar las propiedades del sistema drásticamente, pero el usuario puede no tener manera de saber que se ha producido el cambio. Cada uso del sistema puede encontrarse ejecutándose sobre una versión diferente del software o sobre un ambiente de utilización diferentes (por ejemplo, una configuración de red diferente) para el que no es aplicable la información histórica. La información relativa a la evolución del sistema puede que no esté disponible directamente. En estas situaciones, resulta inadecuado el uso de técnicas tradicionales de inferencia estadística como de soporte en la medición y predicción de los valores de CQA

Para estimar un valor para de un CQA de un sistema en evolución en el que el estado del sistema es inescrutable, se requiere de un método que haga el mejor uso de cada fuente de evidencia relevante disponible, incluida información obtenida de fabricantes, el juicio personal, la historia acumulada, y el conocimiento de la ocurrencia de eventos específicos (actualizaciones a un componente, etc.) que pueda invalidar la información histórica.

En la actualidad, tales aplicaciones basadas en red a menudo son puestas a punto manualmente en base a la intuición y a un conocimiento incompleto del estado del sistema corriente. *La solución CQA reemplaza estas técnicas informales mediante el uso de métodos estadísticos basados en el teorema de Bayes*[20] *que proporcionan una solución basada de manera sistemática en la respuesta para la estimación de valores CQA*. Estos métodos estadísticos proveen un marco matemático que permite la representación de todo aquello que se conocido (o asumido) acerca de un CQA en un formato funcionalmente simple, y sirven de base para la actualización de este

[20] El Teorema de Bayes es un medio de cuantificar la incertidumbre. Basado en la teoría de la probabilidad, el teorema define una regla para el refinamiento de una hipótesis mediante la factorización de evidencia adicional e información de *background*, y conduce a un número que representa el grado de probabilidad de que la hipótesis sea cierta.

formato funcional a partir de nuevos conocimientos de los que se disponga [Lee 89, Royall 97].

Una probabilidad basada en el teorema de Bayes es un formalismo de soporte para razonar acerca de las nociones que se tengan bajo condiciones de incertidumbre y utilizando fuentes de evidencia dispares. A diferencia de la inferencia estadística clásica, esta solución considera que las observaciones se fijan, y que los parámetros son variables aleatorias con sus propias distribuciones estadísticas. La solución comienza con un conjunto adecuado de nociones pre-existentes (la "distribución previa"); a medida que se van disponiendo de nuevos datos, se los combina con la distribución previa para así obtener una distribución nueva (posterior) que puede ser utilizada como base para el análisis.

Cuando no se cuenta con evidencia pre-existente creíble, las "previas" se deben basar en el juicio profesional. Por ejemplo, para un uso de sistema crítico, uno podría emplear valores de CQA previos que representen el peor caso hasta contar con evidencia que indique que la situación es menos severa. (Comenzar con una noción previa de disponibilidad=0 para un servicio, donde disponibilidad es la probabilidad de que el servicio estará disponible cuando se lo necesita, la noción previa se volverá irrelevante de manera creciente, y el valor estimado para la disponibilidad se debería incrementar).

El empleo de la solución basada en el Teorema de Bayes para las actualizaciones de los CQAs ofrece varias ventajas, entre las que caben mencionar:

- Permite tanto predecir y establecer/testear datos que se combinan como una función precisa y conveniente para el CQA.
- Deja disponible todo el conocimiento que se tenga de los valores CQA.
- Permite actualizaciones sencillas del conocimiento de manera apropiada.

Sin embargo, esa solución supone factores de riesgo independientes y, en consecuencia, tiende a sobrestimar el riesgo cuando existen múltiples factores de riesgo correlacionados. Además, la naturaleza dinámica de los CQAs y la posible correlación entre los CQAs puede tornar dificultosa la validación de las mediciones CQA. Para complicar más aún las cosas, para la evaluación funcional de los CQAs, las ejecuciones de flujo se deben tratar como ensayos estadísticamente independientes, ignorando potenciales problemas tales como el estado interno del sistema y los datos almacenados. En consecuencia, debe tenerse cuidado de poder asegurar que:

- Asignar "previas" suficientemente conservadoras.
- Contar con suficiente experiencia antes de incorporarlas a la distribución "posterior"
- Cada función QA se re-evalúa y re-inicializa periódicamente en base a la información histórica y a cualquier conocimiento acerca de cambios en el sistema distribuido.

Esta solución para la actualización dinámica de las proyecciones CQA resulta mucho mejor que

otras alternativas, dado que no resulta posible contar con un conocimiento completo del estado actual y futuro del sistema[21]. Los métodos que se describen aquí, en particular el modelo de atributo funcional que asocia utilización de servicio con valores de atributos, el modelo de transición de estado para los atributos de evaluación, y los métodos basados en el Teorema de Bayes para las actualizaciones de atributos dinámicos, constituyen el marco general de la investigación "Atributos de Calidad Computables". Se requiere de trabajo para desarrollar modelos de atributos específicos dentro de este marco general. El requerimiento de que los atributos sean medibles en una métrica definida a los fines de su cómputo también permite la comprensión humana y el análisis cuando éste no es posible.

6.6 ARQUITECTURAS DE GESTION DE FLUJO

Como se lo indicara anteriormente, las *Estructuras de Flujos* y los *Atributos de Calidad Computables* dan el soporte a arquitecturas de sistemas que realizan una gestión dinámica de flujos y atributos durante la ejecución de los mismos. Las *Arquitecturas de Gestión de Flujo*, FMAs, pueden ofrecer marcos generales para el diseño e implementación con el mencionado propósito, como así también los procesos de ingeniería para el desarrollo de la arquitectura del sistema. Se prevé la existencia de familias abiertas y en evolución de marcos generales de FMA para el desarrollo arquitectónico en el corto y el mediano plazo.

En el corto plazo, las plantillas FMA pueden definir topologías y capacidades funcionales que satisfagan los atributos de calidad para la gestión de flujos localizados. Por ejemplo, un atributo de calidad podría requerir la aislamiento dentro de la red de una clase particular de flujos a los fines de la seguridad. Tal requerimiento a menudo se presenta en el caso de usuarios externos que acceden a los sitios Web de la organización. En este caso, la plantilla FMA podría definir una topología basada en DMZ para aislar los usuarios externos de las redes de la organización, sumado al aislamiento funcional de los servidores Web operativos y de desarrollo; asimismo, los flujos administrativos también podrían aparecer dentro del *Conjunto de Flujos* relevantes, y la plantilla debería incluir las capacidades topológicas y funcionales para el monitoreo y control de la operación del sitio Web. Se debe observar en este ejemplo ilustrativo que los atributos de calidad que comprenden el aislamiento del flujo son impuestos por la organización que lo ejecuta y no por quienes originan el flujo; esto es así en el caso que los servicios de red atravesados por flujos implementen sus propios atributos y procedimientos de gestión para el procesamiento de los flujos entrantes.

En el largo plazo, las plantillas FMA pueden definir las topologías y las capacidades funcionales

[21] Los detalles matemáticos y ejemplos se proporcionan en Walton 02.

para la gestión de las instanciaciones de flujos solicitados por el usuario y la conciliación de requerimientos de atributos en aquellos casos que resulte posible con la capacidad de servicio en la operación en tiempo real; tales plantillas pueden contener proyecciones de tráfico, factores geográficos, patrones de comunicación, y conjunto de otros factores que conducen el diseño de redes de gran alcance.

Los *frameworks* FMA también pueden incluir procesos de ingeniería para el mapeo de especificaciones un *Conjunto de Flujos* dentro de los diseños de red y de servicio. Los *Conjuntos de Flujos* definen los requerimientos de acceso para la conectividad lógica entre los servicios como una topología de red suficiente, como así también los requerimientos funcionales para los servicios en sí mismos. Los atributos de calidad asociados con los *Conjuntos de Flujos* imponen requerimientos y restricciones adicionales al diseño de la red. Estas relaciones entre la utilización de la red embebidas en los flujos y atributos, y el diseño de red embebido en la conectividad y la funcionalidad proporcionan una oportunidad para desarrollar prácticas de ingeniería orientada al desarrollo y validación de una red.

6.7 CONCLUSIÓN

La identificación del flujo, del servicio y de la calidad como conceptos fundamentales para el desarrollo de sistemas de gran escala basados en red resulta importante para lograr la unificación de esta compleja actividad ingenieril. Los fundamentos teóricos desarrollados en este trabajo pueden prescribir prácticas de ingeniería que mejorarán la gestión, adquisición, análisis, desarrollo, operación y evolución de los sistemas. Las siguientes observaciones resumen la visión del trabajo.

- La Ingeniería FSQ facilita la reducción de la complejidad y la mejora de la supervivencia en el desarrollo y operación de sistemas de gran escala basados en red compuestos por cualquier combinación de desarrollos nuevos y componentes COTS.
- La Ingeniería FSQ provee estructuras semánticas sistemáticas e independientes de la escala para los requerimientos, especificación, diseño, verificación, e implementación.
- La Ingeniería FSQ soporta la descomposición transparente partiendo de los flujos de tareas de usuario, servicio y requerimientos de atributos de calidad hasta alcanzar implementaciones de flujos, servicios y atributos de calidad intrínsecamente trazables.
- Los flujos de servicios de usuario y los atributos de calidad permiten del desarrollo del sistema en términos de las visiones a nivel de usuario de los servicios, en oposición a la descomposición o composición basadas en objetos estrictacmente funcionales.
- Las *Estructuras de Flujos* son desterminísticas para la comprensión y análisis de las personas, a pesar del comportamientos asincrónico de la red, con lo que se pueden establecer

procedimientos computacionales para el refinamientos, la abstracción y la verificación.

- Las *Estructuras de Flujos* reflejan las realidades de los sistemas basados en red al tener que enfrentar los *Factores Inciertos*, a fin de soportar la gestión del riesgo de la organización y la supervivencia del sistema.
- Las *Estructuras de Flujos* soportan la definición de flujos de ataque y de intrusión para valorizar las vulnerabilidades y compromisos del sistemas, como una base para las mejoras de la seguridad y la supervivencia.
- Los *Atributos de Calidad Computables* reflejan las realidades de los sistemas basados en red, en la valorización y conciliación de los requerimentos y capacidades de calidad como un proceso intrínsecamente dinámico.
- Los *Atributos de Calidad Computables* proporcionan una visión de la calidad independiente de la escala, centrada en el uso (en lugar de hacerlo centrada en el sistema).
- Las *Arquitecturas de Gestión de Flujos* proveen métodos sistemáticos y uniformes para la gestión de las instanciaciones de los flujos de usuarios y la satisfacción de los atributos de calidad durante la ejecución.
- Los fundamentos de las *Estructuras de Flujos* pueden estimular el desarrollo en el camp de la representación y análisis de flujos en el nivel de los requerimientos en el ámbito de las organizaciones, y en el nivel de la implementación dentro de las arquitecturas de sistemas.
- Los fundamentos de los *Atributos de Calidad Computables* puede estimular la investigación en el ámbito del modelado y la evolución dinámica de importantes atributos y métricas de calidad.

Los esfuerzos en la investigación y el desarrollo de FSQ continuarán a fin de explorar los fundamentos de las *Estructuras de Flujos*, los *Atributos de Calidad Computables* y las *Arquitecturas de Gestión de Flujos*. Se encuentran en preparación varios reportes que detallan los fundamentos matemáticos de los dos primeros [Pleszkoch 02, Walton 02].

SECCION 7: LA PROPUESTA DE UN MODELO DE ATAQUE

Y finalizando con lo planteado en la Propuesta de Tesis[22], habiéndose tomando como base el Plan de Trabajo del CERT en el campo de los Sistemas con Capacidad de Supervivencia., se expuso que:

> *La documentación estructurada y el uso sistemático de patrones de ataque y estrategias de supervivencia pueden ayudar al diseño y el análisis de arquitecturas resistentes a la intrusión. La importante inversión en tecnología de seguridad de la información por parte de una organización a menudo se ve escasamente reflejada, y por ende resulta cuestionada, como un valor para la misión operativa. Una razón importante es que gran parte de los esfuerzos en el diseño y el análisis se centran en la decisión de cuáles tecnologías de seguridad más populares se han integrar, en lugar de realizar una evaluación racional acerca de la manera de enfrentar los ataques que son de esperarse se produzcan para comprometer la misión. El trabajo en equipo incluye la incorporación de técnicas de análisis de intrusión y de riesgo a las prácticas de desarrollo existentes.*

Hacia finales del año 2001, se publica el trabajo "*Attack Modeling for Information Security and Survivability*" realizado por Andrew P. Moore, Robert J. Ellison y Richard C. Linger, en el que se describe e ilustra una aproximación para la documentación de la información de ataque en un formato estructurado y reutilizable. Su objetivo era la de proporcionales a los analistas de seguridad una metodología destinada a documentar e identificar patrones de ataques frecuentes, patrones que, a su vez, fueran empleados por diseñadores y analistas de sistemas de información para desarrollar sistemas de información con mayor capacidad de supervivencia.

El punto de partida era el hecho que muchas de las disciplinas la ingeniería se apoyan en datos de fallas para introducir mejoras en los diseños, práctica que desafortunadamente, es poco corriente en el caso de los ingenieros en sistemas de información, quienes generalmente no utilizan los datos de falla de la seguridad –particularmente los datos de ataques- para mejorar la seguridad y la supervivencia de los sistemas que desarrollan.

Históricamente, la razón de que esto resulte así ha sido que las empresas y las organizaciones son reticentes en revelar información acerca de ataques sufridos por sus sistemas por temor a perder la confianza del público o a que otros atacantes pudiesen abusar de las mismas vulnerabilidades o similares.

[22] Propuesta de Tesis Maestría en Redes de Datos "Análisis de *Survivable Networks* y Evaluación de la Metodología TRIAD", Postulante Ing. Susana C. Romaniz.

Sin embargo, el creciente interés del público y la cobertura de los medios de seguridad en Internet estaban dando por resultado un aumento en la publicación de datos de ataques en libros, grupos de noticias en Internet, y las notificaciones de seguridad del CERT, por ejemplo. De ahí la propuesta orientada a que los ingenieros puedan utilizar estos datos de una manera estructurada para mejorar la seguridad y la supervivencia de los sistemas de información.

7.1 EL PROBLEMA

Los ingenieros en sistemas generalmente no utilizan datos de falla de seguridad –en particular datos sobre ataques- para mejorar sus diseños e implementaciones. Esto se debe, particularmente, a la falta de datos disponibles en forma pública [Anderson 93]. *Las organizaciones privadas y gubernamentales son reticentes a llamar la atención sobre los ataques a sus sistemas por temor a que otros atacantes abusen las mismas o similares vulnerabilidades. Aún luego que sus sistemas han sido reforzados para bloquear ataques, las organizaciones se resisten a divulgar el ataque por temor a perder la confianza del público*.

A pesar de la reticencia de las organizaciones en divulgar los ataques sobre sus sistemas, se han comenzado a conocer datos sobre ataques en mayor medida que durante la pasada década, principalmente por el aumento del interés público y la cobertura de los medios que alienta la seguridad sobre Internet. Organizaciones tales como la *Software Engineering Institute* del *CERT Coordination Center* fueron creadas con la finalidad de ayudar a proteger los sistemas de información de las empresas y los gobiernos de los ataques de seguridad basados en Internet, en parte mediante la publicación de avisos de seguridad que no revelan los nombres de las organizaciones involucradas. Asimismo, se han publicado muchos libros sobre el tema de cómo los *hackers* fuerzan su ingreso en los sistemas.

No obstante, investigaciones recientes demuestran que los ingenieros en sistemas de información no están aprovechando la experiencia derivada de estos ataques de seguridad documentados [Arbaugh 00] *Por ello, los sistemas de información que están siendo construidos y gestionados son pasibles de las mismas o similares vulnerabilidades que los han plagado durante años.*

Los ingenieros en sistemas de información necesitan de una mejor manera de utilizar y analizar los datos de ataques para aprender de la experiencia previa. Este trabajo *propone un medio para documentar los ataques de seguridad de la información en un formato estructurado y reutilizable*. Se espera que los analistas de seguridad sean capaces de utilizar las estructuras descriptas para identificar los patrones de los ataques que ocurren derivados de los datos de ataques re-

ales[23]. Además, es deseable que los diseñadores y analistas de sistemas de información sean capaces de utilizar los patrones de ataque para desarrollar sistemas con mayor capacidad de supervivencia.

La solución de documentación se basa en una estructura denominada ***árbol de ataque*** (*attack tree*) [Schneier 99]. En las siguientes secciones se describen el formato y la semántica de árbol de ataque, una estructura para los patrones genéricos de captura y reutilización de los ataques de seguridad de la información, y, finalmente, un modelo para el refinamiento de los árboles de ataque basado en la especificación y reutilización de estos patrones de ataque genéricos.

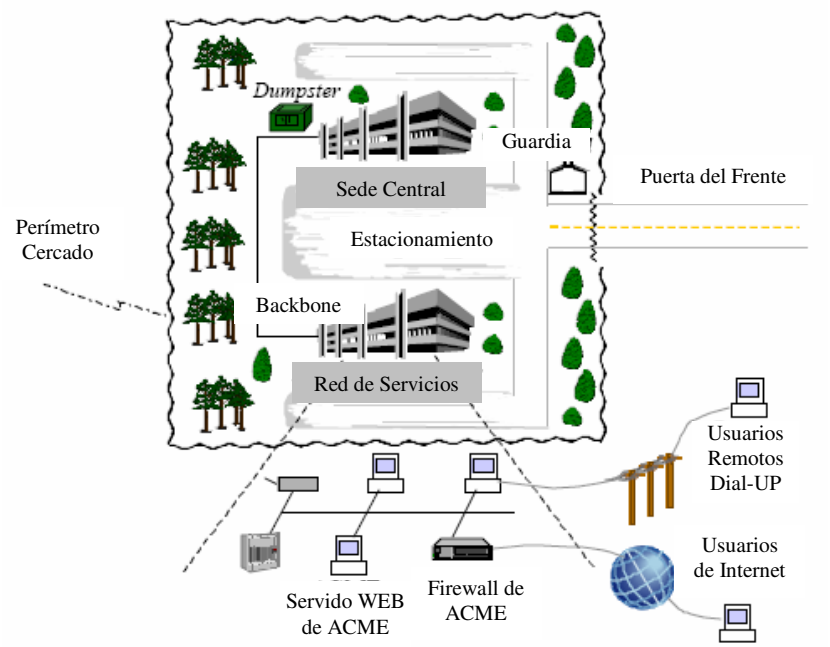


Figura 38. ACME, Inc. Arquitectura de la empresa.

A lo largo del trabajo se utilizan ataques sobre una compañía ficticia, llamada ACME, Inc., para ilustrar conceptos y problemas. La Figura 38 muestra el ambiente y la arquitectura de la empresa ACME. Las características importantes para destacar son que las propiedades de ACME están protegidas por una cerca alrededor del perímetro. La única entrada a la propiedad es a través del perímetro cercado. Además del cerco perimetral, la seguridad física consta de una puerta al frente custodiada. Las redes locales están divididas entre la LAN de la Sede Central y la LAN de la Red de Servicios. Los usuarios de Internet se conectan al Servidor ACME a través de un firewall. Los usuarios vía enlaces telefónicos conmutados ganan acceso a un servidor particular sobre la LAN de la Red de Servicios.

[23] Estas estructuras no están pensadas para ser utilizadas por las víctimas de un ataque, sino por los analistas quienes comprenden más plenamente los perfiles del atacante y el impacto de los ataques específicos sobre la operación del sistema.

7.2 ÁRBOLES DE ATAQUE

Los árboles de ataque han existido de variadas formas, y bajo diversos nombres, desde hace muchos años, pero el formato más recientemente publicado los describe como *un método sistemático para caracterizar la seguridad de un sistema en base a ataques diversificados* [Schneier 00a]. *Refinan información acerca de ataques mediante la identificación del compromiso de la seguridad o la supervivencia de la organización indicada como la raíz del árbol. Las maneras en que un atacante puede causar este compromiso iterativa e incrementalmente son representadas como nodos de menos nivel del árbol*. Una organización generalmente posee un conjunto, o ***bosque de árboles de ataque*** (*forest*) que son relevantes para su operación. *La raíz de cada árbol del bosque representa un evento que podría dañar de manera significativa la misión de la organización. Cada árbol de ataque enumera y elabora las maneras en las que un atacante podría hacer que el evento ocurra. Cada camino a lo largo de un árbol de ataque representa un ataque único sobre la organización.*

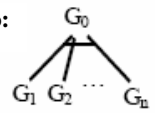
7.2.1 Estructuras y semánticas

Un nodo de un árbol de ataque se descompone de una de las dos siguientes maneras:

- *Un conjunto de sub-objetivos de ataque, todos los cuales deben ser alcanzados para que el ataque tenga éxito, que están representados como una descomposición-AND*, o
- *Un conjunto de sub-objetivos de ataque, alguno de los cuales debe ser alcanzado para que el ataque resulte exitoso, que están representados como una descomposición-OR*.

Los ataques se pueden representar gráficamente o textualmente. Una ***descomposición-AND*** se representa de la siguiente manera:

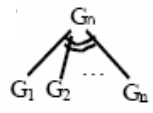
Gráfico:



Texto: **Objetivo** G_0
AND G_1
G_2
...
G_n

representa un objetivo G_0 que puede ser alcanzado si el atacante logra cada uno de los sub-objetivos de G_1 a G_n. De manera similar se representa una ***descomposición-OR***:

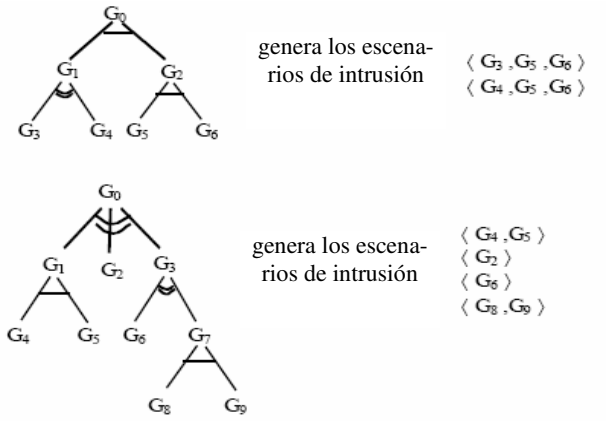
Gráfico:



Texto: **Objetivo** G_0
OR G_1
G_2
...
G_n

representa un objetivo G_0 que puede ser alcanzado si el atacante logra uno cualquiera de los sub-objetivos de G_1 a G_n. Generalmente, en el trabajo se utiliza una representación textual debido a

que la representación gráfica tiende a ser dificultosa para el caso de árboles de ataque no triviales. *Los árboles de ataque constan de alguna combinación de descomposiciones –AND y –OR*. Generamos escenarios de intrusión particulares a partir de un árbol de ataque recorriendo el árbol en forma descendente. Por ejemplo,



En general, se agregan hojas de objetivos al final de los escenarios a medida que van siendo generados. *Las descomposiciones-OR hacen que se creen nuevos escenarios. Las descomposiciones-AND hacen que se extiendan los escenarios existentes. Los nodos intermedios del árbol de ataque no aparecen en los escenarios de intrusión debido a que están elaborados por los objetivos de menor nivel.*

Los árboles de ataque permiten el refinamiento de los ataques a un nivel de detalle elegido por el desarrollador. ***Poseen la propiedad de transparencia referencial***, tal como fuera caracterizada por Prowell:

> "*La transparencia referencial implica que los detalles relevantes de menor nivel de una entidad son abstraídos en lugar de ser omitidos dentro de la descripción de mayor nivel un sistema particular, de tal manera que la descripción de nivel superior contiene todo lo que sea necesario para la comprensión de la entidad cuando se la coloca dentro de un contexto mayor*" [Prowell 99].

Esta propiedad permite que el desarrollador explore ciertos caminos de ataque con mayor profundidad que otros, y seguir permitiéndole generar escenarios de intrusión que tengan sentido. Además, el refinamiento de las ramas del árbol de ataque genera nuevas ramas, dando por resultado escenarios de intrusión con un menor nivel de abstracción.

7.2.2 Arbol de ataque de ACME

La Figura 39 ejemplifica un árbol de ataque de alto nivel en el que el nodo raíz comprometa a la seguridad de ACME con la divulgación de secretos de propiedad. Observar que se han incluido

tanto ataques físicos y de ingeniería social como ataques tecnológicos. La rama 1 de la descomposición del nivel más alto trata de los ataques denominados *dumpster-diving*, tanto en el lugar como luego que los mismos han sido retirados.

Compromiso de supervivencia: divulgación de los secretos de propiedad privada de ACME

OR 1. Escarbar en la basura ítems físicos descartados de ACME
- ***OR*** 1. Inspeccionar el contenido de los contenedores de basura in situ
- 2. Inspeccionar la basura luego de retirada del lugar

2. Monitorear las emanaciones de las máquinas ACME
- ***AND*** 1. Relevar el perímetro físico para determinar la posición óptima de monitoreo
- 2. Adquirir el equipo de monitoreo necesario
- 3. Establecer el lugar de monitoreo
- 4. Monitorear las emanaciones desde el sitio

3. Reclutar ayudar de alguien de adentro ACME confiable
- ***OR*** 1. Infiltrar al espía como alguien de adentro confiable
- 2. Utilizar al espía interno

4. Acceder físicamente a las redes o máquinas ACME
- ***OR*** 1. Obtener acceso físico in situ a la Intranet
- 2. Obtener acceso físico a las máquinas externas

5. Atacar la Intranet ACME utilizando sus conexiones con Internet
- ***OR*** 1. Monitorear las comunicaciones sobre Internet por alguna filtración
- 2. Acceder a un proceso de confianza para enviar información al atacante sobre Internet
- 3. Ganar acceso privilegiado al servidor Web

6. Atacar la Intranet ACME utilizando sus conexiones con la red de telefonía pública
- ***OR*** 1. Monitorear las comunicaciones sobre la red de telefonía pública por alguna filtración de información sensible
- 2. Ganar acceso privilegiado a las máquinas sobre la Intranet conectada vía Internet

Figura 39. Árbol de ataque de alto nivel para ACME.

La rama 2 elabora los ataques que monitorean las emanaciones (por ejemplo, visual o electromagnética) desde sitios exteriores próximos al perímetros. Las ramas 3 y 4 se refieren a ataques que abusan las personas internas confiables y el acceso físico (ya sea local o remoto), respectivamente. Finalmente, las ramas 5 y 6 caracterizan los ataques tecnológicos sobre la Internet y sobre la red de telefonía pública. Se han considerado los ataques que abusan debilidades técnicas y no-técnicas de una operación crítica para la empresa a fin de fortalecer la seguridad y la supervivencia de la información [Schneier 00a, Anderson 93].

Si bien los escenarios de intrusión para el árbol de ataque de la Figura 39 son de muy alto nivel, se los lista para ilustrar las propiedades de transparencia referencial de los árboles de ataque. Cabe recordar que los nodos intermedios no se incluyen en un escenario de intrusión debido a que están completamente elaborados por los nodos de menor nivel. Se utiliza la notación (i, j, k) para representar el escenario de intrusión de la hoja objetivo i, seguido por el paso j, y seguido por el paso k.

- (1.1), (1.2), (2.1, 2.2, 2.3, 2.4), (3.1), (3.2)

- (4.1), (4.2), (5.1), (5.2), (5.3), (6.1), (6.2)

La mayoría de estos escenarios son de longitud 1 debido a que no son parte de una descomposición-AND. La única excepción es la descomposición-AND bajo la rama 2 de la Figura 39.

Ahora se supone, por ejemplo, que se necesita conocer más acerca de los ataques a través de la Internet sobre el Servidor Web de ACME (es decir, la rama 5.3 de la Figura 39). La Figura 40 elabora ataques sobre el Servidor Web de ACME que tiene por objetivo ganar acceso privilegiado.

5.3 Ganar acceso privilegiado al servidor Web de ACME

AND 1. Identificar el nombre de dominio de ACME
2. Identificar la dirección IP del *firewall* de ACME
OR 1. Interrogar al servidor de nombre de dominio
2. Escanear la identificación del *firewall*
3. Trazar la ruta a través del *firewall* hasta el servidor Web
3. Determinar el control de acceso del *firewall* de ACME
OR 1. Buscar puertos particulares en estado *listening* por omisión
2. Escanear intensivamente puertos para detectar cualquier puerto en estado *listening*
4. Identificar el sistema operativo y tipo del servidor Web
OR 1. Escanear *banners* de servicios de SO para la identificación el SO
2. Realizar sondeos sobre el *stack* TCP/IP para obtener información sobre las características del SO
5. Abusar de las vulnerabilidades del servidor Web
OR 1. Acceder directamente a recursos sensibles compartidos a través de la Intranet
2. Acceder a datos sensibles desde la cuenta privilegiada del servidor Web

Figura 40. Refinamiento del ataque al servidor Web

Los escenarios para este sub-árbol que permite acceder a recursos compartidos de la Intranet sensibles en forma directa (es decir, 5.3.5.1) son los siguientes (nota: se ha omitido el prefijo 5.3 en la etiqueta de cada hoja para simplificar):

- (1, 2.1, 3.1, 4.1, 5.1), (1, 2.2, 3.1, 4.1, 5.1), (1, 2.3, 3.1, 4.1, 5.1)
- (1, 2.1, 3.2, 4.1, 5.1), (1, 2.2, 3.2, 4.1, 5.1), (1, 2.3, 3.2, 4.1, 5.1)
- (1, 2.1, 3.1, 4.2, 5.1), (1, 2.2, 3.1, 4.2, 5.1), (1, 2.3, 3.1, 4.2, 5.1)
- (1, 2.1, 3.2, 4.2, 5.1), (1, 2.2, 3.2, 4.2, 5.1), (1, 2.3, 3.2, 4.2, 5.1)

El conjunto de escenarios para ganar acceso a datos sensibles desde una cuenta protegida sobre el servidor Web (5.3.5.2) es idéntico al conjunto anterior en el que se ha sustituido 5.2 por 5.1. Estos 24 escenarios podrían reemplazar el escenario (5.3) de la lista original de escenarios de intrusión para el árbol de ataque de la Figura 38. Esta lista representa los escenarios de intrusión para el árbol de ataque refinado.

7.2.3 Reutilización del patrón de ataque

El uso práctico de los árboles de ataque para caracterizar los ataques que sufren los sistemas del mundo real depende de la posibilidad de la reutilización de los patrones de ataque previamente desarrollados. A continuación se describen dos estructuras que permiten tal reutilización: *un patrón para caracterizar un tipo de ataque particular, y un perfil de ataque para la organización de los patrones de ataque de tal manera que resulte más fácil su búsqueda y su aplicación.*

7.2.3.1 Patrones de ataque

Se define a un ***patrón de ataque*** como *una representación genérica de un ataque malicioso e intencional que ocurre de manera común en contextos específicos*. *Cada patrón de ataque contiene:*

- *el objetivo global del ataque especificado por el patrón*
- *una lista de pre-condiciones para utilizarlo*
- *los pasos para llevar adelante el ataque*
- *una lista de post-condiciones que son verdaderas si el ataque resulta exitoso.*

Las pre-condiciones incluyen las suposiciones que se hicieron acerca del atacante o del estado de la empresa que son necesarias para que el ataque tenga éxito; ejemplo de pre-condiciones incluyen las destrezas, recursos, accesos, o conocimientos que el atacante posee, y el nivel de riesgo que está dispuesto a tolerar. Las post-condiciones incluyen el conocimiento ganado por el atacante y los cambios en el estado de la empresa que resultan de la realización de los pasos del ataque cuando se dan las pre-condiciones.

En el Anexo 6 se presentan un conjunto de ejemplos de utilización de patrones de ataque.

7.2.3.2 Perfiles de ataque

Luego se organizan los patrones de ataque relacionados dentro de un ***perfil de ataque*** abarcativo. *Los perfiles de ataque contienen:*

- *un modelo de referencia común*
- *un conjunto de variantes*
- *un conjunto de patrones de ataque*
- *un glosario de términos y frases definidas*

El modelo de referencia representa una plantilla de arquitectura con parámetros que pueden incluir variantes específicas. Los patrones de ataque también están definidos en términos de estas variantes. Como describiremos de manera más completa seguidamente, los patrones de ataque se especifican de manera independiente de cualquier empresa en particular. Una empresa

cuya arquitectura es consistente con un modelo de referencia del perfil puede utilizar los patrones de ataque del mismo, una vez instanciado, para ayudar a construir los árboles de ataque relevantes para la operación de la empresa. Los diferentes perfiles de ataque pueden incluir diferentes niveles de acceso, recursos y destrezas del atacante, como así también diferentes configuraciones de los componentes del sistema. En consecuencia*, los diferentes patrones de ataque pueden ayudar a refinar un árbol de ataque específico de una empresa a lo largo de las diferentes líneas de ataque.*

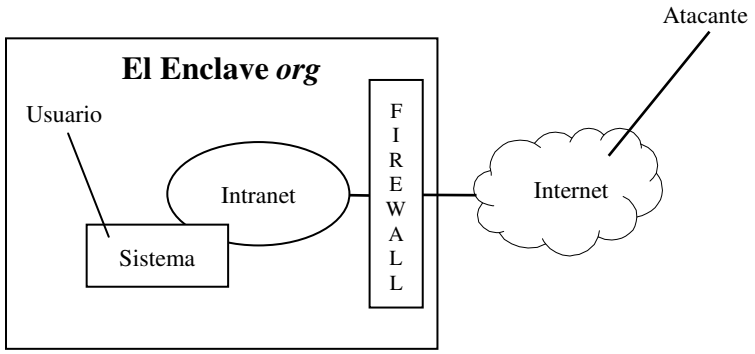


Figura 41. Modelo de Referencia del Ataque de Enclave Basado en Internet

La Figura 41 describe un modelo de referencia para el Perfil de Ataque de Enclave Basado en Internet. Las variantes de este modelo de referencia son los términos indicados en itálicas en la figura: *Usuario, Sistema, Intranet, Firewall, Atacante*. Es decir, estos elementos pueden variar dependiendo de los detalles de la empresa específica. Los patrones de ataque también pueden estar especificados en términos de éstas, y potencialmente, otras variantes.
En el Anexo 6, también se muestra un ejemplo de utilización de un patrón de ataque.

7.3 REFINAMIENTO DEL ARBOL DE ATAQUE

Como se muestra en el diagrama de flujo de la Figura 42, *un árbol de ataque puede ser refinado desde el nodo raíz mediante una combinación de extensiones manuales y aplicaciones de patrones. Las extensiones manuales dependen particularmente de la experiencia de la persona que está desarrollando el árbol de ataque. La aplicación de patrón también depende de tal experiencia, pero en menor grado. Parte de esta experiencia se encuentra embebida en la librería de patrones de ataque.* De aquí en adelante, se asume que existe esta librería.

Una *librería de patrón de ataque provee un conjunto de perfiles de ataque que son lo suficientemente rico como para caracterizar los ataques que pueden tener lugar en un amplio rango de arquitecturas de organizaciones*. El refinamiento de un árbol de ataque particular comprende el descubrimiento de aquellos perfiles de ataque que son consistentes con la arquitectura de la

empresa. El desarrollador investiga los patrones de ataque de los perfiles de ataque consistentes para un refinamiento de un camino de ataque contenido en el árbol de ataque de la empresa. Una vez encontrado, el desarrollador puede instanciarlo de manera apropiada y aplicar el patrón de ataque a fin de extender el árbol de ataque de la empresa. Este proceso de entremezclado de aplicación de patrón con extensión manual continúa hasta que el árbol de ataque se haya refinado suficientemente.

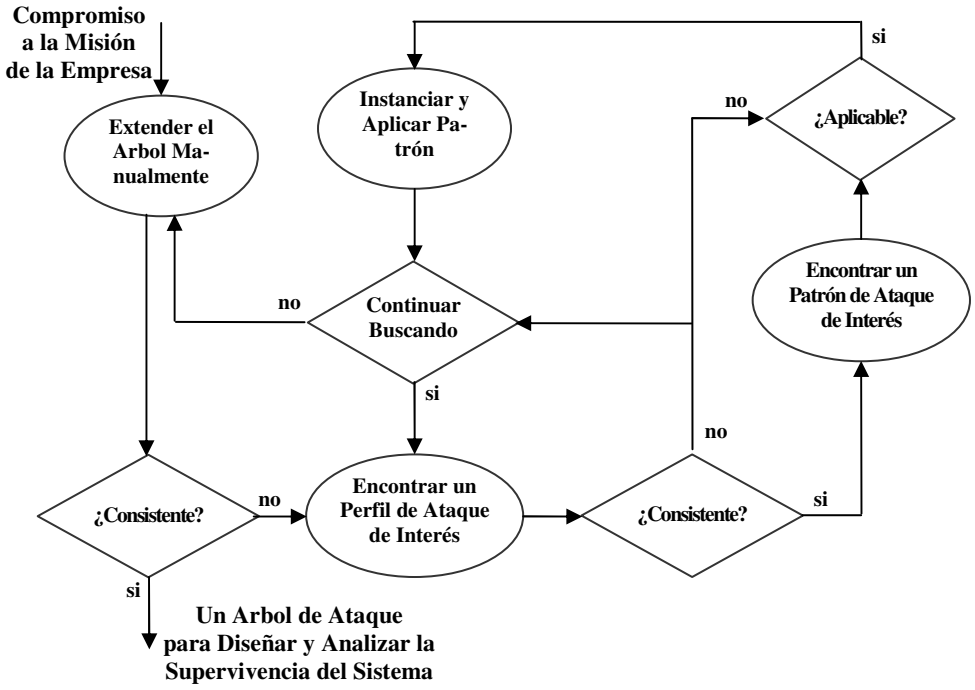


Figura 42. Proceso de refinamiento del árbol de ataque.

Una *librería de patrón de ataque provee un conjunto de perfiles de ataque que son lo suficientemente rico como para caracterizar los ataques que pueden tener lugar en un amplio rango de arquitecturas de organizaciones*. El refinamiento de un árbol de ataque particular comprende el descubrimiento de aquellos perfiles de ataque que son consistentes con la arquitectura de la empresa. El desarrollador investiga los patrones de ataque de los perfiles de ataque consistentes para un refinamiento de un camino de ataque contenido en el árbol de ataque de la empresa. Una vez encontrado, el desarrollador puede instanciarlo de manera apropiada y aplicar el patrón de ataque a fin de extender el árbol de ataque de la empresa. Este proceso de entremezclado de aplicación de patrón con extensión manual continúa hasta que el árbol de ataque se haya refinado suficientemente.

A continuación se analiza con mayor detalle qué significa que un perfil de ataque sea consistente con una arquitectura de empresa, qué significa que un patrón de ataque sea aplicable al árbol de ataque de la empresa, y de qué manera instanciar y aplicar un patrón de ataque para refinar el

árbol de patrón de ataque de la empresa. La decisión de cuándo detener el proceso queda a criterio del desarrollador.

7.3.1 Consistencia Perfil/Empresa

Como se mencionara anteriormente, el modelo de referencia asociado con un perfil de ataque puede ser visto como una plantilla de arquitectura. Los parámetros de esta plantilla son las variantes del modelo de referencia. *Si existe un conjunto de valores para estas variantes que unifican el modelo de referencial del perfil de ataque con alguna porción de la arquitectura de la empresa, se dice que el modelo es* ***consistente con*** *la arquitectura de la empresa.* Los patrones de ataque asociados con el perfil se escriben con respecto al modelo de referencia del perfil y en términos de las variantes del perfil. En consecuencia, estos patrones de ataque son relevantes para la arquitectura de la empresa.

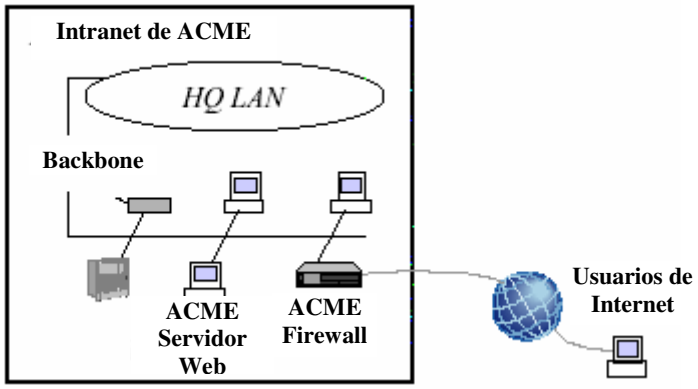


Figura 43. Intranet de la Empresa ACME

Como un ejemplo específico de la consistencia de un perfil de ataque con la arquitectura de la empresa, veamos el Perfil de Ataque de Enclave Basado en Internet descripto anteriormente. El modelo de referencia para este perfil, ilustrado en la Figura 41, es consistente con la arquitectura de la empresa ACME, que se muestra en la Figura 38. Esto se puede ver mediante la instanciación de las variantes del perfil que se indica: *Org* como ACME, *Intranet* como Intranet de ACME, *Firewall* como firewall de ACME. Si bien la Intranet de ACME no está rotulada explícitamente como tal en la Figura 38, se la puede caracterizar como se muestra en la Figura 43. Las restantes variantes asociadas con el modelo de referencia permanecen abstractas, representando simplemente *Uuario*, *Atacante* y *Sistema* que se instancian en una etapa posterior del refinamiento.

Desde luego, se podrían utilizar otros perfiles de ataque para refinar el árbol de ataque de ACME, tal como el Perfil de Ataque de Enclave Basado en PTN. Este perfil contiene patrones de ataque sobre la red de telefonía pública vía enlaces telefónicos. El modelo de referencia para este perfil, que se muestra en la Figura 44, es vulnerable al Ataque de Sondeo por Discado:

Patrón de Ataque de Sondeo de Discado:

Objetivo: Loguearse en forma remoto al *Sistema*

Pre-condición:

1. *Atacante* posee información sobre la central telefónica de la *Org*
2. El *Atacante* conoce el nombre de usuario del *Usuario* en el *Sistema*

Ataque:

AND 1. Sondear la central telefónica de *Org* para detectar modems que responden
2. Determinar que se realiza la conexión al *Sistema* a través del *MODEM*
3. Loguearse al *Sistema* como *Usuario*

Post-condición: El *Atacante* tiene acceso a la cuenta del *Usuario* sobre el *Sistema.*

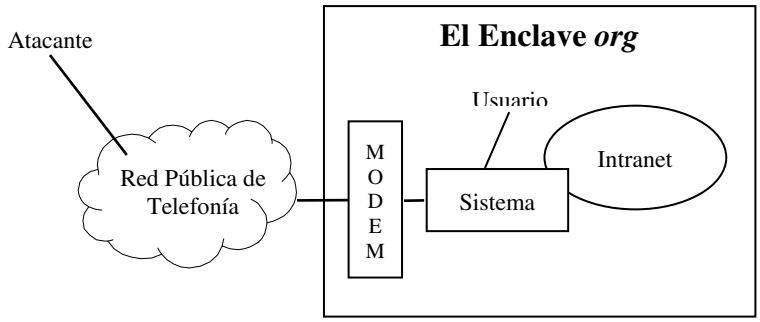


Figura 44. Modelo de Referencia del Ataque de Enclave Basado en RTP

7.3.2 Aplicación del Patrón

La determinación de cuáles perfiles de ataque son consistentes con la arquitectura de la empresa es sólo el primer paso. Los analistas también deben determinar cuáles patrones de ataque dentro de los perfiles consistentes ayudan a refinar el árbol de ataque de la empresa. *Esto requiere de la identificación de un patrón cuyo objetivo ayude a alcanzar el objetivo identificado en un nodo del árbol de ataque. Se dice que tales patrones, cuando han sido adecuadamente instanciados, son* ***aplicables*** *al árbol de ataque de la empresa.*

Por ejemplo, supongamos que se necesita utilizar el Patrón de Ataque de Desbordamiento de Buffer, definido en el Anexo 6, para refinar el árbol de ataque de ACME de la Figura 38. Observamos que un atacante podría alcanzar el objetivo 5.3.5.2 (es decir, Acceso a datos sensibles desde una cuenta privilegiada sobre el servidor Web de ACME) al tener acceso a dicha cuenta y luego desencadenar una búsqueda de archivos que contienen datos sensibles.

Asimismo, el atacante podría alcanzar el primero de estos sub-objetivos mediante el abuso de la vulnerabilidad de desbordamiento de buffer en el servidor Web de ACME. Pero esto se parece al objetivo de Patrón de Ataque de Desbordamiento de Buffer.

Se puede utilizar el Patrón de Ataque de Desbordamiento de Buffer si se lo instanciara de tal manera que el *Sistema* bajo ataque sea el Servidor Web de ACME y la *función maliciosa* que se esté ejecutante le permita al atacante tener acceso a la cuenta privilegiada:

Patrón de Ataque de Desbordamiento de Buffer: (instanciado)

Objetivo: Abusar la vulnerabilidad de desbordamiento de buffer para obtener acceso a una cuenta privilegiada sobre el Servidor Web de ACME.

Pre-condición: El *Atacante* puede ejecutar ciertos programas sobre el Servidor Web de ACME

Ataque:

AND 1. Identificar un programa ejecutable sobre el Servidor Web de ACME susceptible de la vulnerabilidad de desbordamiento de buffer.

2. Identificar el código que podría proporcionar acceso a una cuenta privilegiada cuando se lo ejecuta con el privilegio del programa.
3. Construir el valor de entrada que forzará a que el código se ubique en el espacio de direcciones del programa.
4. Ejecutar le programa de tal manera que haga que el mismo salte a la dirección en la cual reside el código.

Post-condición: El *Atacante* puede acceder a la cuenta privilegiada.

5.3.5.2 Acceso a datos sensibles desde una cuenta privilegiada en en Servidor Web de ACME
AND 1. Obtener acceso a una cuenta privilegiado sobre el Servidor Web de ACME
AND 1. Identificar un programa ejecutable sobre el Servidor Web de ACM susceptible de la vulnerabilidad de desbordamiento de buffer
2. Identificar el código que podría proporcionar acceso a la cuenta privilegiada cuando se lo ejecuta con el privilegio del programa
3. Construir el valor de entrada que forzará a que el código se ubique dentro del espacio de direcciones del programa.
4. Ejecutar el programa de tal manera que lo haga saltar a la dirección donde reside el código.
2. Analizar archivos en la búsqueda de datos sensitivos.

Figura 45. Refinamiento del ataque de desbordamiento de buffer.

No se requiere una rutina de sustitución de las instanciaciones, sino más bien reformularlas de tal manera que suenen naturales y preserven la intención original del patrón. La Figura 45 muestra una porción del árbol de ataque refinado como resultado de la aplicación del patrón de ataque anterior. En consecuencia, se puede refinar el árbol de ataque de ACME de tal manera que permita el uso del patrón. *Este refinamiento dirigido del árbol de ataque de la empresa aplicado a un patrón de ataque específico es una forma común de permitir la reutilización.*

La Figura 46 ilustra los tres diferentes tipos de aplicación de patrón. Cada fila indica el árbol de ataque que resulta de la aplicación de un patrón de ataque, con una instanciación genérica, a un

tipo de nodo particular de un árbol de ataque de empresa. Los árboles y patrones de ataque que carecen de un significado de descomposición AND u OR pueden ser una descomposición AND o una OR. Por ejemplo, la aplicación de un patrón de ataque al nodo hoja de un árbol de ataque de la empresa, que se muestra en la primera fila de la Figura 46, no depende de si el nodo o patrón son descompuestos AND u OR. No obstante, sí depende de una instanciación del patrón que permite que el nodo hoja sea refinado.

Se presenta la instanciación de un patrón de ataque como la instanciación del objetivo de cada patrón, lo que se denota de manera abstracta como un "*i*" seguido por el objetivo del nodo instanciado. En consecuencia, aplicado al nodo hoja, la instanciación del objetivo del patrón (iG_R) debe lograr que el nodo hoja sea redefinido (G_{K+i}). El refinamiento del árbol de ataque de ACME utilizando el Patrón de Ataque de Desbordamiento de Buffer, que se muestra en la Figura 45, ejemplifica una aplicación de patrón a un nodo hoja.

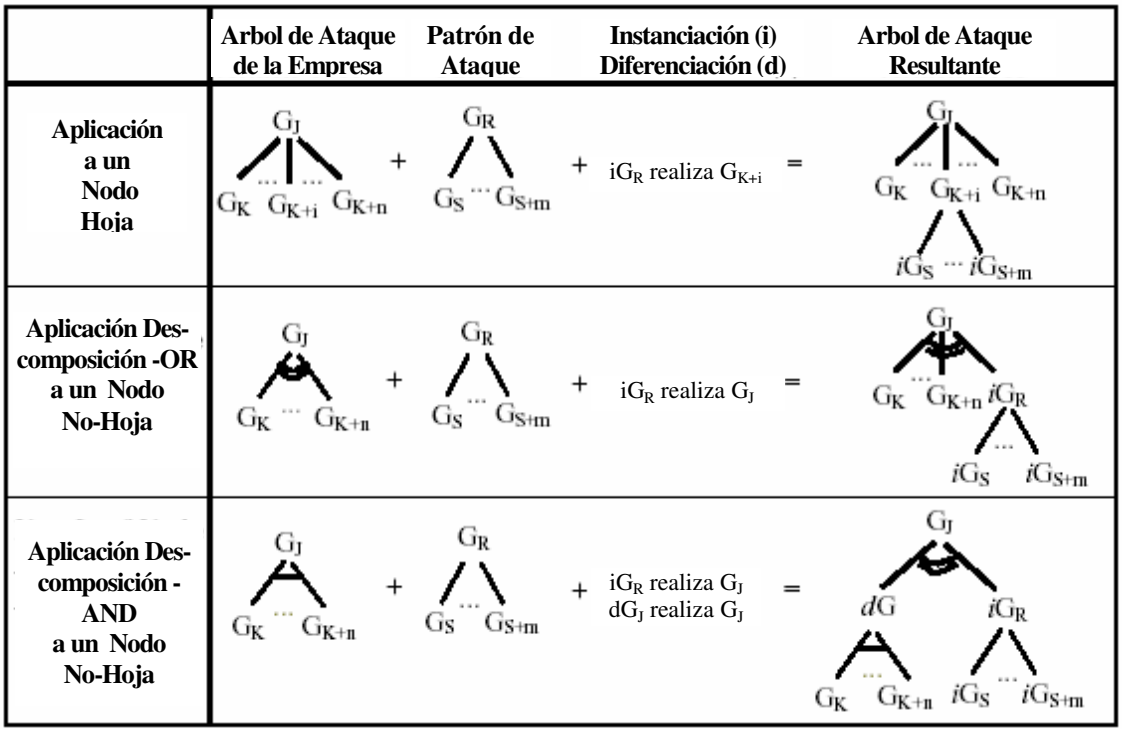


Figura 46. Aplicación de Patrones de Ataque

Las aplicaciones en nodos no-hoja se muestran en la segunda y tercera filas de la Figura 46 dependen de si el nodo está descompuesto AND u OR. La aplicación de un patrón de ataque a un nodo descompuesto OR, como se muestra en la segunda fila, simplemente resulta en otra rama OR que se adiciona al árbol de ataque en ese nodo. Esta rama OR representa otro camino para que el atacante logre el objetivo G_J, uno por medio del ataque original y otro por medio del patrón de ataque instanciado. En este caso, el usuario debe diferenciar el ataque original respecto del nuevo

patrón de ataque. El objetivo diferenciado está representado en forma abstracta como G_J.

Para ilustrar la aplicación de un patrón de ataque a un nodo intermedio del árbol de ataque, se debería instanciar el Patrón de Ataque de Operador Inesperado descripto en Anexo 6, al igual que se hizo con el Patrón de Ataque de Desbordamiento de Buffer. El patrón resultante, donde el *Sistema* bajo ataque es el Servidor Web ACME y la *función maliciosa* que se ejecuta le otorga al atacante acceso a una cuenta privilegiada, es como sigue:

5.3.5.2 Acceso a datos sensible desde cuenta privilegiada sobre servidor Web de ACME

AND 1. Tener acceso como cuenta privilegiada sobre el Servidor Web de ACME

OR 1. Abusar la vulnerabilidad de desbordamiento de *buffer* para acceder como cuenta privilegiada

AND 1. Identificar el programa ejecutable sobre el Servidor Web de ACME susceptible de la vulnerabilidad de desbordamiento de *buffer*

2. Identificar el código que podría proporcionar acceso como cuenta privilegiada cuando se ejecuta con el privilegio del programa

3. Construir el valor de entrada que forzará a que el código se encuentre en el espacio de direcciones del programa

4. Ejecutar el programa de tal manera que salte a la dirección donde reside el código

2. Abusar la vulnerabilidad de operador inesperado para acceder como cuenta privilegiada

AND 1. Identificar el programa ejecutable sobre el Servidor Web de ACME susceptible de la vulnerabilidad de operador inesperado

2. Identificar el operador (inesperado) que permite la composición de la llamada al sistema

3. Identificar la llamada al sistema que podría proporcionar acceso como cuenta privilegiada cuando se ejecuta con el privilegio del programa

4. Construir una entrada inesperada componiendo un valor de entrada legal con la llamada al sistema utilizando el operador inesperado

5. Ejecutar el programa sobre el Servidor Web de ACME con la entrada inesperada

2. Acceder a datos sensibles desde la cuenta privilegiada del servidor Web

Figura 47. Refinamiento del Ataque de Operador Inesperado.

Patrón de Ataque de Operador Inesperado: (instanciado)

Objetivo: Abusar la vulnerabilidad de Operador Inesperado para tener acceso como una cuenta privilegiada.

Pre-condición: El *Atacante* puede ejecutar ciertos programas sobre el Servidor Web de ACME.

Ataque:

AND 1. Identificar en el Servidor Web de ACME un programa ejecutable susceptible de la vulnerabilidad de operador inesperado.

2. Identificar el operador (inesperado) que permite la composición de llamadas al sistema.

3. Identificar la llamada al sistema que podría tener acceso como una cuenta pri-

vilegiada cuando se ejecute con el privilegio del programa.

4. Construir la entrada inesperada componiendo el valor de la entrada legal con la llamada al sistema que emplea el operador inesperado.
5. Ejecutar el programa sobre el Servidor Web de ACME con la entrada inesperada.

Post-condición: El Atacante puede acceder como la cuenta privilegiada.

La Figura 47 muestra el árbol de ataque de la Figura 45 refinado para aplicar este patrón en el nodo 5.3.5.2.1. La tercera fila de la Figura 46 representa este tipo de aplicación de patrón de ataque. Se debe observar que se ha utilizado el objetivo del Patrón de Ataque de Desbordamiento de Buffer como el objetivo diferenciado.

7.4 CONCLUSIONES

El objetivo de este trabajo es describir un medio de documentación de información de ataques de seguridad de una manera estructurada y reutilizable. Dentro de su alcance, se ha mostrado cómo documentar posibles ataques sobre la empresa en el formato de árboles de ataque. Cada árbol de ataque enumera y elabora las maneras en las que un atacante puede comprometer la capacidad de la empresa de cumplir con su misión. Se describe cómo documentar y organizar patrones genéricos de ataque y cómo reutilizarlos para facilitar la construcción del árbol de ataque. Los diferentes ejemplos provistos ilustran las estructuras y técnicas empleadas.

Del mismo probablemente surjan más preguntas que respuestas a ellas. Por ejemplo:

- ¿De qué manera se pueden derivar requerimientos y mejorar los diseños de sistema a partir de ataque conocidos?
- ¿Qué tipos de análisis se pueden realizar sobre los árboles de ataque?
- ¿Con qué nivel(es) de detalle se deberían caracterizar los patrones de ataque?
- ¿Existe un lenguaje más estructura para los patrones de ataque que facilite su combinación y análisis?
- ¿Cómo se maneja la volatilidad del descubrimiento de vulnerabilidades y del *patching* de sistema?
- ¿De qué manera se pueden priorizar las ramas de un árbol de ataque de acuerdo con la probabilidad e impacto?
- ¿Cómo se puede determinar la habilidad y motivación del atacante en ejecutar ataques particulares?

Existen algunas preguntas que orientan la investigación futura. El objetivo global de esta investigación es el desarrollo de métodos para derivar requerimientos y diseños de los sistemas y operaciones que soporten de mejor manera los ataques activos y maliciosos. Se considera que la

incorporación de las lecciones aprendidas de los ataques previos es un aspecto importante de esta investigación y que el árbol de ataque es una estructura útil para organización los datos históricos de ataques. Como líneas de trabajo se tienen:

- refinar la metodología que facilite un análisis más sistemático
- validar la practicabilidad y escalabilidad de la metodología
- desarrollar un amplio rango de perfiles de ataque que soporten reutilización
- formalizar el modelo de refinamiento y análisis del árbol de ataque
- determinar el rol de automatización apropiado para soportar esta metodología.

SECCION 8: CONCEPTOS COMUNES EN EL AMBITO DE LA INGENIERIA DE LA SEGURIDAD, LA PROTECCIÓN Y LA SUPERVIVENCIA

Como último aspecto a analizar, y de acuerdo a lo planteado en la Propuesta de Tesis[24], en esta sección se considera el trabajo realizado por Donald G. Firesmith "*Common concepts underlying safety, security, and survivability engineering* "[Firesmith 03e], publicado a finales del año 2003.

Si bien la sociedad descansa en los sistemas basados en software, y confía vidas, bienes, e incluso el entorno en la exitosa operación de estos sistemas basados en tecnología, esto *no son ni perfectos ni invulnerables*. Generalmente fallan debido a defectos en el software, a fallas en el hardware, al uso accidental inadecuado, y al abuso deliberado; también son objeto de ataques maliciosos por parte de hackers, empleados descontentos, criminales, espías industriales, terroristas, e incluso agentes de gobiernos y militares extranjeros. Pero su posible falla debería cada vez más estar bajo control en la medida en que cada vez más dependemos de ellos.

Por este motivo, la ingeniería en los campos de la protección, de la seguridad y de la supervivencia se está volviendo componentes esenciales de la ingeniería en sistemas. Estas tres disciplinas estrechamente relacionadas se beneficiarán significativamente en la medida que se avance en el amplio reconocimiento de sus similitudes y diferencias. Todavía hoy, quienes se desenvuelven en ellas tienden a establecer una inadecuada interacción, tanto entre sí como con miembros de otras disciplinas de la ingeniería de sistemas de información.

Esta inadecuada interacción resulta especialmente cierta en la ingeniería de requerimientos aplicada al campo de protección, seguridad y supervivencia con respecto a cuáles son los componentes arquitectónicos asociados en los que se deberían basar (por ejemplo, prácticas de seguridad tales como capacitación y procedimientos, contramedidas de seguridad tales como encriptación y firewalls, y mecanismos de protección tales como redundancia y componentes de seguridad).

Es más, *por lo general los ingenieros de requerimientos en protección, seguridad y supervivencia emplean terminologías diferentes* [CNSS 03, van der Meulen 00] *y procesos que enfatizan sus diferencias y oscurecen sus similitudes. A menudo, el resultado son especificaciones de requerimientos muy incompletas en lo referente a protección, seguridad y supervivencia, que carecen de características fundamentales como verificabilidad y ausencia de ambigüedad*.

Si bien los requerimientos de protección, seguridad y supervivencia están comenzando ha ser objeto de interés, generalmente *se carece de la fundamentación teórica para definir qué es lo que*

[24] Propuesta de Tesis Maestría en Redes de Datos "Análisis de *Survivable Networks* y Evaluación de la Metodología TRIAD", Postulante Ing. Susana C. Romaniz.

son y de qué manera se encuentran relacionados con otros conceptos esenciales, ya que hasta hace poco, el énfasis ha estado especialmente puesto sobre los componentes arquitectónicos y la evaluación de la protección, seguridad y supervivencia de los sistemas y arquitecturas existentes.

El análisis y la especificación de los requerimientos de protección, seguridad y supervivencia son dificultosos por naturaleza. A diferencia de otros requerimientos que especifican la capacidad requerida (y deseada), ***estos requerimientos especifican qué deben prevenir*** (es decir, accidentes y ataques debidos a peligros para la protección o amenazas a la seguridad). Los mismos tienen que ver con los activos que se deben proteger y con la administración de los riesgos de daño a estos activos. A su vez, deberían ser apropiados y posibles; no tiene sentido especificar un requerimiento cuyo valor de implementación sea muy superior al valor del daño sufrido por el activo (y de cualquier otro activo que de manera subsecuente pudiera ser dañado).

Por otra parte, ***existe un nivel inherente de incertidumbre debido a cuáles eventos de los que estos requerimientos intentan prevenir pueden o no suceder alguna vez, y el grado de impacto que presentan***. Esto resulta particularmente cierto con los requerimientos de protección debido a que algunos sistemas, tales como plantas de energía nuclear y plantas químicas, son tan críticos que aún un único y excepcional accidente puede conducir al sistema a una falla total; en cambio otros sistemas, tales como sitios Web de comercio electrónico, se encuentran bajo constantes ataques, pero al presentar una misión crítica menor, el daño debido a amenazas de seguridad y a un ataque exitoso no conduzce al sistema a una condición similar.

Otro problema es **que** ***los peligros y amenazas asociadas con los sistemas de basados en software están constantemente cambiando***, *haciendo que resulte muy difícil cuantificar los riesgos; a menudo, las estimaciones de los riesgos son conjeturas y, en consecuencia, los riesgos generalmente, por fuerza, cualitativos en lugar de cuantitativos*.

Atendiendo a que actualmente se pueden distinguir tres disciplinas ligadas a la ***protección*** –*safety*– grado en que se previene, detecta y reacciona ante un *daño accidental*, la ***seguridad*** –security– grado en que se previene, detecta y reacciona ante un daño malicioso, y la ***supervivencia*** –survivability– grado en que se previene, detecta y reacciona ante tanto un daño *accidental* como uno *malicioso* sobre un servicio esencial, y a que estas disciplinas presentan tanto similitudes como diferencias, las que resulta necesario distinguir a los fines de poder establecer patrones de calidad en cuanto a los requerimientos asociados a una estrategia de supervivencia, se lleva a cabo una reseña del trabajo mencionado al inicio de la sección.

8.1 ANALISIS DE LAS REDES CON CAPACIDAD DE SUPERVIVENCIA

La ingeniería en el campo de la *protección*, de la *seguridad* y de la *supervivencia* son tres disci-

plinas estrechamente relacionadas que se podrían beneficiar significativamente a partir del reconocimiento de sus similitudes y diferencias. Sin embargo, actualmente los actores de estas disciplinas interactúan inadecuadamente, tanto entre sí como con los actores de otras disciplinas de la ingeniería. Esta inadecuada interacción se verifica en relación con *los requerimientos* a partir de los cuales se deberían establecer los componentes arquitectónicos asociados (por ejemplo, protección y prevención). Los ingenieros en requerimientos y en protección, seguridad y supervivencia por lo general emplean terminologías y procesos diferentes que acentúan sus diferencias y oscurecen sus similitudes. Muy frecuentemente, el resultado es que ***las especificaciones de requerimientos son muy incompletas en lo que hace a protección, seguridad y supervivencia. Los "requerimientos" que se especifican generalmente carecen de características necesarias tales como verificabilidad y ausencia de ambigüedad***.

A esto se suma que la mayoría de los libros y artículos acerca de protección, seguridad y supervivencia no describen de manera adecuada los requerimientos para la especificación mínima y exigible de estos factores de calidad. Es decir, no tratan la adquisición, el análisis y la especificación de tales requerimientos, ni orientan sobre cómo deberían verse los mismos. *En lugar de discutir cómo especificar el nivel de protección, seguridad y supervivencia que resulta necesario, discuten accidentes y ataques, peligros y amenazas, riesgos, vulnerabilidades, y los componentes arquitectónicos que utilizan para prevenir, detectar o reacción ante estos peligros y amenazas*. De hecho, la mayor parte del material publicado en esta área ***ni hace mención de la ingeniería de requerimientos asociada***. Esto resulta ser especialmente cierto con los libros de seguridad, tal vez porque ***los ingenieros en seguridad tienden a pensar en términos de políticas de seguridad en lugar de requerimientos de seguridad*** [Alberts 03, Herrmann 99, Hughes 95, McNamara 03, Peltier 01, Power 00, Schneier 00a, Shema 03, Tulloch 03], y cuando hacen referencia a requerimientos generalmente sólo proporcionan una reseña breve y de alto nivel, sin decir claramente cuáles son estos requerimientos [Anderson 01, Leveson 95, McDermid 91]**. *Pero sin los adecuados requerimientos basados en políticas y en objetivos, ¿cómo podemos estar seguros que los mecanismos de protección, seguridad y supervivencia que hemos seleccionado nos han de proteger de adecuada o apropiada***?

El trabajo en consideración presenta un *conjunto de modelos de información* aplicados al campo de interés que proveen un fundamento básico para la ingeniería de protección, de seguridad y de supervivencia, comenzando con un resumen del ***concepto del modelo de seguridad y sus componentes***, tres de los cuales son los ***factores de calidad***: protección, seguridad y supervivencia. Seguidamente, describe los diferentes tipos de requerimientos, y proporciona ***un framework que muestra los objetivos, políticas y requerimientos***, en el que relaciona los componentes arquitectónicos con los factores y sub-factores de calidad (criterios y métricas) del modelo de

calidad. Finalmente, resume el modelo general, presenta tres ***modelos que relacionan conceptos de protección, seguridad y supervivencia, y donde describe las estrechas relaciones entre estos tres campos de la ingeniería***. Además de los modelos de información gráficos, se proporcionan ***definiciones precisas para cada concepto***.

Como conclusión, ***establece las similitudes y diferencias entre los modelos que subyacen a estas tres disciplinas de la ingeniería***. *Las similitudes entre ellas superan significativamente sus diferencias*; son importantes porque les permiten a los ingenieros desarrollar procesos relativamente uniformes, los cuales pueden ser utilizados de manera transversal por las tradicionalmente separadas ingenierías en protección, seguridad y supervivencia.

A menudo, las ingenierías en protección, seguridad y supervivencia no están adecuadamente reconocidas como disciplinas fuertemente relacionadas. La protección está particularmente relacionada con la atención a los daños de activos de valor (en especial personas) debidos a accidentes. La seguridad, con la atención a los daños de activos de valor (en especial datos sensibles) debidos a ataques. Y la supervivencia, con la atención a los daños de activos de valor (servicios esenciales) ante tanto accidentes como ataques. *En los tres casos, el foco principal está puesto en los peligros y amenazas y sus riesgos asociados y en las vulnerabilidades del sistema frente a ellos. Las tres disciplinas a menudo requieren de una metodología basada en el riesgo para la determinación de las políticas, requerimientos y componentes arquitectónicos adecuados*.

Sin embargo, estas similitudes son pocas veces reconocidas, y su pertinencia en lo que hace a la ingeniería de requerimientos es en gran medida desconocida en la práctica actual. El problema es que *los ingenieros en requerimientos raramente abordan temas relativos con la ingeniería en protección, la ingeniería en seguridad o alguno relacionado con supervivencia, y muy pocas veces poseen alguna experiencia significativa en estas disciplinas.* No obstante, a menudo son responsables de desarrollar requerimientos de protección, seguridad y supervivencia. Un buen punto de partida para ellos es comenzar a aprender los fundamentos de protección, seguridad y supervivencia. De manera similar, *los ingenieros en protección y seguridad rara vez conocen sobre ingeniería en requerimientos y generalmente no poseen experiencia en este campo. Tienden a pensar en términos de planificación de la protección y de políticas de seguridad más que en requerimientos y sus especificaciones.* Además, las políticas que producen están a un nivel superior de abstracción que los requerimientos y, por lo general, carecen de las características de buenos requerimientos tales como integridad, ausencia de ambigüedad y verificabilidad.

8.1.1 Modelo de Calidad

Calidad significa mucho más que simplemente satisfacer requerimientos funcionales. Aún en el caso que una aplicación proporcione todas las características requeridas y satisfaga cada uno

de sus casos de uso, la misma todavía puede ser totalmente inaceptable si sus atributos de calidad resultan insuficientes (por ejemplo, ofrece una inadecuada disponibilidad, su capacidad es demasiado baja, su desempeño es demasiado bajo, si no tiene capacidad de interoperar con otros sistemas, si su uso no resulta seguro, si posee numerosas vulnerabilidades, o si los usuarios la consideran poco "amigable"). En consecuencia, *el término "calidad" es un término abstracto que puede significar cosas muy diferentes para los diferentes participantes (incluidos clientes, usuarios, administradores, fabricantes, desarrolladores, testadores, ingenieros de calidad, encargados del mantenimiento, y personal de soporte)*. En el presente trabajo, el término "calidad" se emplea en su sentido más amplio. En consecuencia, ***calidad incluye todos los factores de calidad y no sólo los mencionados anteriormente***.

De manera similar, resulta inadecuado especificar la calidad requerida estableciendo simplemente que una aplicación deberá poseer alta capacidad y desempeño, ser confiable y segura, y ser usable por sus usuarios finales. Tales "requerimientos de calidad" por lo general ***se especifican a tal alto nivel de abstracción que los mismos resultan prácticamente inútiles debido a que son incompletos, vagos, ambiguos, e imposibles de verificar***. Es decir, resultan ser objetivos a alto nivel más que requerimientos específicos.

En consecuencia, *es necesario descomponer el término "calidad" en sus componentes relevantes. También se necesita proporcionar definiciones operativas para estos componentes de calidad de tal manera que se puedan crear requerimientos claros, no ambiguos y verificables*.

Para especificar los requerimientos de calidad se necesita organizar, clarificar, y estandarizar los significados relevantes del término "calidad" cuando se lo aplica a los sistemas basados en software. Esto permitirá contar con una fundamentación apropiada para identificar, analizar y especificar el gran número de requerimientos de calidad que resultan necesarios para cualquier iniciativa significativa.

Un ***modelo de calidad*** es un modelo (es decir, una colección de abstracciones o simplificaciones relacionadas) que modela la calidad de algo [Firesmith 03a]. El modelo de calidad hace al concepto de calidad lo que un mapa hace a una ciudad, una provincia o un país; capta todos los aspectos generales importantes acerca de la calidad, en tanto que ignora todos los detalles distractivos. Éste es, entonces, el ***rol de un modelo de calidad: hacer específico y de utilidad el término general "calidad" descomponiéndolo en sus componentes constitutivos y las relaciones entre ellos***. *Un primer modelo de calidad descompone la calidad en sus factores de calidad constitutivos (aspectos, atributos o características) y sus sub-factores (es decir, partes). De esta manera, el mismo provee criterios (descripciones) y métricas (medios de medición) específicas de calidad que se pueden utilizar para transformar estos factores de calidad generales y de alto nivel en descripciones detalladas y mensurables que se pueden utilizar para especificar un as-*

pecto de calidad o para determinar si ese aspecto de calidad realmente existe en un nivel igual o superior al mínimo especificado en una especificación de requerimientos. Al exigir una combinación de criterio y métrica de calidad, aumenta la posibilidad de obtener una declaración clara, no ambigua y verificable de un requerimiento de calidad.

Existen muchos modelos de calidad que difieren en los niveles de detalle y de facilidad de uso. Algunos son estándares internacionales [ISO 00], otros son estándares de facto en la industria [Firesmith 03d], otros son específicos de las organizaciones [Barbacci 00], y otros están publicados en libros de ingeniería de software [Boehm 76, Chung 93, Chung 00, Davis 93, Loucopoulos 95, Mylopoulos 92, Roman 85, Sommerville 92, Thayer 90]. El presente trabajo emplea el modelo de calidad *Open Process, Environment, and Notation* (*OPEN*) [Firesmith 03d] debido a su nivel de detalle, especialmente en vista a protección, seguridad y supervivencia.

8.1.1.1 Modelo de Información para un Modelo de Calidad

Como se ilustra en la Figura 48, ***un modelo de calidad es un modelo jerárquico compuesto por factores de calidad (también conocidos como atributos de calidad), los cuales contienen sub-factores de calidad***. ***El modelo formaliza el concepto de calidad, y descompone la calidad en una clasificación de factores y sub-factores de calidad relevantes. Para cada uno de ellos, define el criterio y las métricas de calidad asociadas que proporcionan las descripciones específicas mensurables que están siendo analizadas o especificadas***.

En la Figura 48 se muestra que los factores y sub-factores de calidad son las principales vías en las que se descompone el concepto de calidad. Sin embargo, *la calidad se vuelve específica en consideración a los sistemas cuando se la juzga en base a un criterio de calidad específico de la aplicación y se la mide en términos de métricas de calidad específicas*.

Dicha figura es un diagrama de clase UML que documenta el meta-modelo de un modelo de calidad (donde un meta-modelo es, en realidad, un modelo de un modelo). *El meta-modelo para el modelo de calidad define las clases de cosas que constituyen cualquier modelo de calidad, mientras que un modelo de calidad contiene los factores, sub-factores, criterios y métricas de calidad reales*. En consecuencia, para el ejemplo, el meta-modelo contiene el concepto de "factor de calidad", mientras que el modelo de calidad contiene los factores de calidad específicos tales como protección, seguridad y supervivencia.

Los conceptos documentados en la Figura 48 se pueden definir de la siguiente manera:

- **Modelo de calidad** es un modelo de calidad jerárquico para formalizar el concepto de calidad en términos de sus factores y sub-factores de calidad constitutivos. Conforma una taxonomía de las características que constituyen la calidad en la que sus componentes deberían ser disjuntos y cubrir todo el dominio objeto de la taxonomía. En conse-

cuencia, los factores de calidad de un modelo de calidad deberían ser disjuntos y abarcar toda la calidad; desgraciadamente, la ingeniería de protección, seguridad y supervivencia se han desarrollado y evolucionado de manera relativamente independiente una de otra, por lo que tienden a superponerse. Las potenciales concordancias entre las tres pueden, incluso, demostrar que se podrían beneficiar con la adopción de conceptos de las otras. Por ello, algunas definiciones de seguridad incluyen daño accidental que de manera más correcta caen dentro de protección. De manera similar, algunas definiciones de protección son incompletas porque sólo incluyen el daño sobre las personas y no incluyen el daño sobre la propiedad o sobre el entorno.

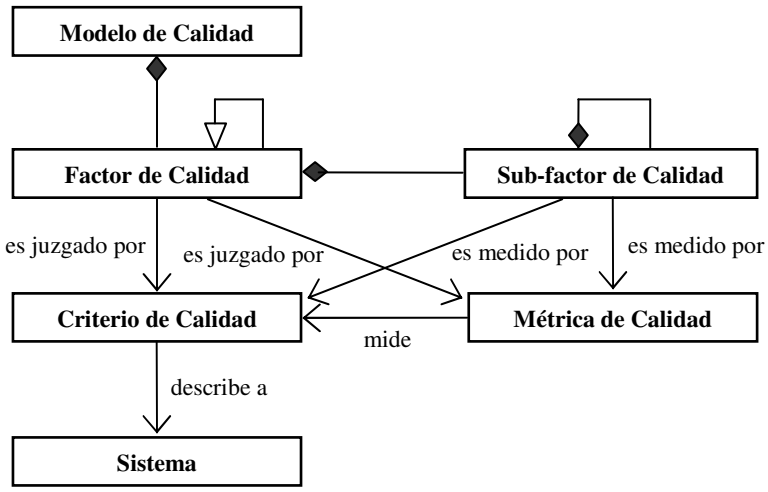


Figura 48. Meta-modelo de información para modelo de calidad.

- **Factor de calidad** (también conocido como atributo de calidad o característica de calidad) es una característica o atributo de alto nivel de algo que capta un aspecto de su calidad. La calidad tiene que ver con el grado en que algo posee una combinación de características, atributos, aspectos o rasgos que son deseables para sus partes implicadas. Existen muy diferentes factores de calidad tales como disponibilidad, extensibilidad, desempeño, fiabilidad, reusabilidad, protección, seguridad y usabilidad. Estos factores determinan si algo posee la suficiente calidad o no. Debido a que muchos de los factores de calidad presentan la terminación "bilidad", a menudo se los conoce como las "bilidades". Los factores de calidad pueden ser sub-clasificados en *categorías de factores de calidad mucho más específicos* (por ejemplo, la fiabilidad es una categoría de confiabilidad). Los factores de calidad también se pueden descomponer en sus *partes constitutivas* (por ejemplo, privacidad como una parte de seguridad).
- **Sub-factor de calidad** es un constituyente principal de un factor o de otro sub-factor de calidad.

- **Criterio de calidad** es una descripción específica de algo que provee evidencia a favor o en contra de la existencia de un factor o sub-factor específico de calidad. Un criterio de calidad contribuye de manera significativa en hacer los factores de calidad de alto nivel lo suficientemente detallados como para que resulten no ambiguos y verificables. Cuando un criterio de calidad es adecuadamente específico, no requiere del agregado de métricas de calidad para hacerlo lo suficientemente completo y detallado como para conformar la base de requerimientos de calidad detallados. Existen muchos más criterios de calidad que factores y sub-factores de calidad debido a que generalmente existen criterios por factor. Los criterios de calidad también son más específicos del dominio y menos reutilizables que los factores y sub-factores de calidad debido a que son descripciones específicas de aplicaciones de negocio. A menudo, para afrontar tan importante número de criterios y hacerlos reutilizables, los criterios de calidad están parametrizados dentro de los modelos de calidad, y las instancias específicas de las clases de criterios parametrizados se pueden utilizar para producir los requerimientos de calidad [Firesmith 03b].
- **Métrica de calidad** es una métrica que cuantifica un criterio de calidad y, en consecuencia, lo hace mensurable, objetivo y no ambiguo. Una métrica de calidad es una manera de medición que cuantifica un criterio de cualidad; en consecuencia las métricas de calidad proporcionan valores numéricos que especifican o estiman la calidad de un producto o de un proceso midiendo el grado en que el mismo posee un factor o un sub-factor de calidad específico.
- **Sistema** (también conocido como sistema a nivel de aplicación) es una colección integrada de componentes de datos, de hardware, de software, de rol humano (también conocido como personal), y de documentos que colaboran en la provisión de un cierto conjunto cohesivo de funcionalidades con niveles específicos de calidad.

8.1.1.2 Una taxonomía de Factores y Sub-factores de Calidad

Como se estableciera anteriormente, un factor de calidad (también conocido como atributo de calidad o característica de calidad) es una característica o atributo de alto nivel correspondiente a alguna cosa que capta un aspecto de su calidad. Debe observarse que los factores y sub-factores de calidad simplemente *describen capacidades deseadas o existentes*. Como tales, *proveen un fundamento y una organización para discutir objetivos, políticas, requerimientos y componentes arquitectónicos de calidad que satisfacen estos requerimientos*. Sin embargo, no son objetivos, políticas, requerimientos o componentes arquitectónicos en sí mismos.

Diferentes autores descomponen los factores de calidad de distinta manera (por ejemplo, factores cualitativos vs. factores cuantitativos). No obstante, si bien puede resultar difícil hacer que todos

los requerimientos de calidad sean cuantitativos, resulta sin lugar a dudas posible y útil lograrlo (por ejemplo, para su testeabilidad). En consecuencia, este trabajo los descompone en *factores de calidad orientados al desarrollo y orientados al uso*. Una taxonomía así parece ser muy intuitiva debido a que separa intereses de acuerdo a sus audiencias (orientado al desarrollo para desarrolladores y encargados del mantenimiento, y orientado al uso para clientes y usuarios).

Esta taxonomía también es relativamente completa. Si bien algunos pocos proyectos necesitan desarrollar requerimientos para todos los factores de calidad indicados en las dos listas siguientes, los ingenieros de requerimientos deberían, sin duda alguna, determinar la aplicabilidad de todos los diferentes tipos de factores de calidad.

El hecho de contar con una extensa taxonomía jerárquica de factores y sub-factores de calidad también ayuda a ignorar factores de calidad que no son relevantes.

La siguiente taxonomía de factores de calidad [Firesmith 03d] sirve a *dos funciones primordiales*:

- Prove un contexto para la protección, seguridad y supervivencia, es decir, proporciona una taxonomía jerárquica dentro de la cual la protección, la seguridad y la supervivencia deben ajustarse lógicamente.
- Mediante la documentación de los factores de calidad más importantes se pone en evidencia que un modelo de calidad completo puede tener un gran número de factores y sub-factores de calidad.

Los *factores de calidad orientados al desarrollo* son particularmente importantes durante el desarrollo y el mantenimiento más que durante el uso. En la Tabla 14 se listan algunos ejemplos de factores y sub-factores de calidad.

Los *factores de calidad orientados al uso* resultan particularmente importantes luego de la implantación y durante el empleo real de una aplicación o componente. En la Tabla 15 se listan algunos ejemplos de factores y sub-factores de calidad orientados al uso.

Esta taxonomía pone en claro que la protección, seguridad y supervivencia son clases de fiabilidad y que, en consecuencia, son factores de calidad orientados al uso. Asimismo, enfatiza que *la protección, la seguridad y la supervivencia son sólo tres de un conjunto de muchos otros factores de calidad potencialmente relevantes*.

8.1.2 La Protección como un Factor de Calidad

El modelo de información indicado en la Figura 57 muestra que la protección es una clase de fiabilidad y, por ello, una clase de factor de calidad. También muestra que la protección ha sido tradicionalmente clasificada en protección de la salud, de la propiedad y del ambiente (también clases de factores de calidad) sobre la base del tipo de activo que resultaría dañado en caso que ocurriera un accidente.

Mantenibilidad	Facilidad con que una aplicación o un componente puede ser mantenido entre dos versiones.
Corregibilidad	Facilidad con que se pueden corregir defectos menores entre versiones mientras la aplicación o el componente está siendo usado por sus usuarios.
Extensibilidad	Facilidad con que, en el futuro, se puede mejorar una aplicación o un componente a fin de satisfacer requerimientos u objetivos cambiantes.
Portabilidad	Facilidad con que una aplicación o componente puede ser movido de un entorno a otro.
Re-usabilidad	Facilidad con que una aplicación o un componente existente puede ser re-utilizado.
Escalabilidad	Facilidad con que una aplicación o un componente puede ser modificado para expandir sus capacidades actuales.
Verificabilidad	Facilidad con que una aplicación o un componente puede ser verificado en cuanto a su capacidad de satisfacer sus requerimientos y estándares asociados.
Testeabilidad	Facilidad con que una aplicación o un componente facilita la creación y ejecución de testeos eficaces (es decir, testeos que pudieran provocar fallas debido a defectos subyacentes.

Tabla 14. Factores de calidad asociados al desarrollo.

Auditabilidad	Grado con el que se mantienen suficientes registros que den soporte a una auditoría del sistema.
Capacidad	Número mínimo de cosas (por ejemplo, transacciones, almacenamiento) que pueden ser manejados de manera exitosa.
Configurabilidad	Grado con el que algo puede ser configurado de múltiples maneras (es decir, configuraciones).
Internacionalización	(también conocido como globalización y localización) Grado con el que algo puede ser o estar configurado apropiadamente para ser usado dentro de un entorno global.
Personalización	Grado con el cual cada usuario particular puede ser presentado con una experiencia única específica del usuario.
Variabilidad	Grado con el que algo existe como múltiples variantes, cada una con las capacidades apropiadas.
Corrección	Grado con el cual un producto de trabajo y sus salidas se encuentran libres de defectos una vez que el producto de trabajo ha sido distribuido.
Exactitud	Magnitud de defectos (es decir, el desvío de las mediciones reales o promedio respecto de sus valores verdaderos) en los datos cuantitativos.
Validez	Grado con el cual los datos se mantienen válidos (es decir, actualizados, no obsoletos).

Precisión	Grado de dispersión de los datos cuantitativos, independientemente de su exactitud.
Fiabilidad	Grado con el cual los diferentes tipos de usuarios pueden depende de un producto de trabajo
Disponibilidad	Grado con el cual un producto de trabajo está operativo y disponible para su uso. El tema de la disponibilidad es la principal dificultad dado que cuando los ingenieros de sistemas y de software piensan en los requerimientos de disponibilidad, lo hacen en términos de causas no-maliciosas que provocan la falta de disponibilidad.
Confiabilidad	Grado con el cual un producto de trabajo sin sufrir fallas bajo condiciones dadas durante un período de tiempo dado.
Robustez	Grado con el cual un producto de trabajo ejecutable continúa funcionando apropiadamente bajo condiciones o circunstancias anormales.
Tolerancia ambiental	Grado con el cual un producto de trabajo ejecutable continúa funcionando adecuadamente a pesar de la existencia de un ambiente anormal.
Tolerancia a error	Grado con el cual un producto de trabajo ejecutable continúa funcionando apropiadamente a pesar de la presencia de una entrada errónea.
Tolerancia a falla	Grado con el cual un producto de trabajo ejecutable continúa funcionando apropiadamente a pesar de la ocurrencia de fallas, donde: • Una falla es la ejecución de un defecto que causa una inconsistencia en el comportamiento real de un producto de trabajo ejecutable real (observado) y el esperado (especificado). • Un defecto puede o no causar una falla dependiendo de si el defecto es ejecutado o y si el manejo de excepciones evita la ocurrencia de la falla. • Un defecto (*fault* o *bug*) es una debilidad subyacente dentro de un producto de trabajo (es decir, un producto de trabajo que es inconsistente con sus requerimientos, políticas, objetivos, o en las expectativas razonables de sus clientes o usuarios). Los defectos por lo general son causados por errores humanos, y no poseen impacto hasta que causan una o más fallas.
Protección	Grado con el cual se previene, reduce o reacciona apropiadamente ante un daño *accidental*.
Seguridad	Grado con el cual se previene, reduce o reacciona apropiadamente ante un daño *malicioso*; el término "malicioso" se utiliza intencionalmente para diferenciar claramente la protección de la seguridad y, de esta manera, evitar una superposición innecesaria dentro de la taxonomía de los factores de calidad; por lo tanto, la protección tiene que ver con accidentes, mientras que la seguridad está relacionada con ataques. No obstante, los accidentes (protección) pueden ser el resultado de vulnerabilidades en la seguridad que pueden ser explotadas por los ataques, en cuyo caso sus consecuencias han de quedar encuadradas dentro del dominio de la seguridad. De manera similar, los ataques pueden causar peligros para la protección, los que a su vez pueden derivar en ser causa de accidentes.

Eficiencia	Grado con el cual algo hace uso de manera efectiva (es decir, minimiza su consumo) de sus recursos. Estos recursos pueden incluir todo tipo de recursos tales como informático (hardware, software y red), maquinaria, facilidades y personal.
Interoperabilidad	Grado con el cual un sistema o uno de sus componentes se encuentra apropiadamente conectado y opera con algún otro.
Compatibilidad operacional con el ambiente	Grado con el cual un sistema o un componente puede ser utilizado y funciona de manera correcta bajo condiciones específicas del o los ambientes dentro de los cuales está destinado a operar.
Desempeño	Grado con el cual resultan adecuadas las características de coordinación con el tiempo (*timing*).
Jitter	Precisión (es decir, variabilidad) con respecto al tiempo con que ocurren uno o más eventos.
Latencia	Tiempo requerido para proveer un servicio solicitado o permitir el acceso a un recurso.
Tiempo de respuesta	tiempo empleado para empezar a responder a una solicitud por un servicio o el acceso a un recurso.
Capacidad de planificación	Grado con el cual se pueden planificar eventos y comportamientos y que los mismos ocurran tal como hayan sido planificados.
Productividad	Número de veces que un servicio puede ser provisto dentro de una unidad de tiempo especificada.
Utilidad	Grado con el cual algo puede ser accedido y utilizado por sus diferentes tipos de usuarios.
Accesibilidad	Grado con el cual la interfase de usuario de algo le permite a los usuarios con discapacidades comunes o específicas (por ejemplo, auditivas, visuales, motrices o cognoscitivas) realizar sus tareas especificadas.
Instalabilidad	Facilidad con la cual algo puede ser instalado de manera exitosa en su ambiente(s) de producción.
Operabilidad	Grado con el cual algo le permite a sus operadores la realización de sus tareas en concordancia con el manual de operaciones.
Facilidad de transporte	Facilidad con la cual algo puede ser movido físicamente desde un lugar a otro.
Usabilidad	Facilidad con la cual los miembros de un grupo especificado de usuarios son capaces de utilizar algo de manera efectiva.
Capacidad de remoción	Facilidad con la cual una versión problemática del sistema o de uno de sus componentes puede ser removida exitosamente y reemplazada por una versión anterior que estuviera operativa.

Tabla 15. Factores de calidad asociados al uso.

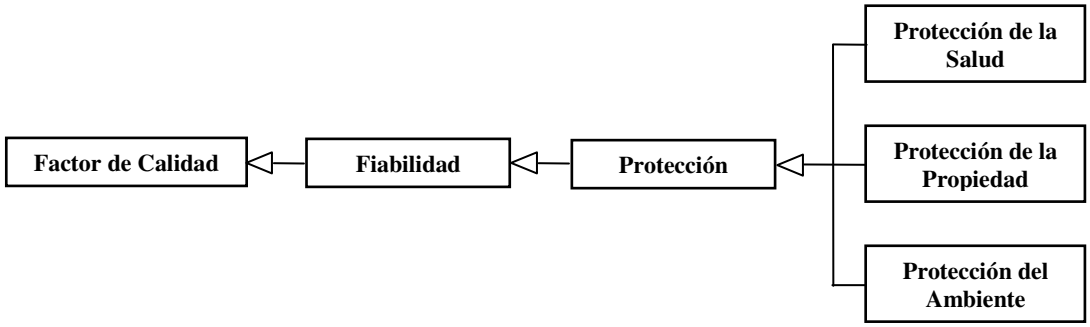


Figura 49. Protección como un Factor de Calidad.

Los conceptos indicados en la Figura 49 se pueden definir de la siguiente manera:

- **Protección** Grado con el cual se previene, reduce o reacciona apropiadamente ante un daño *accidental*. En el uso corriente, las personas no se consideran protegidas a menos que lo estén ante daños tanto accidentales como maliciosos. Además, a menudo la seguridad se entiende que incluye la seguridad ante daños accidentales. Sin embargo, cuando se trata con sistemas, la protección pone el énfasis en los daños accidentales, y la seguridad, en los daños maliciosos. Por lo tanto, para tener una taxonomía de calidad con factores de calidad disjuntos, la protección estará restringida a los daños accidentales, y la seguridad lo estará a los daños maliciosos.

  El factor de calidad de protección se puede clasificar en las siguientes sub-clases de protección, los cuales también son en sí mismos factores de calidad:

  - **Protección de la salud** Grado con que se previene, detecta y se reacciona adecuadamente ante a enfermedad, lesión y muerte. La protección de la salud alcanza a todas las personas que puede esperarse, de manera razonable, sean damnificadas por el sistemas durante un accidente. (Por ejemplo, la protección de la salud correspondiente a un sistema de control automotriz puede incluir al conductor, los pasajeros, los transeúntes y los mecánicos; en el caso de un sistema de control de una planta química, puede incluir operadores, ingenieros de mantenimiento, otro personal de la planta, y los residentes del vecindario).
  - **Protección de la propiedad** Grado con que se previene, detecta o reacciona adecuadamente ante al daño o la destrucción accidentales de la propiedad.
  - **Protección del ambiente** Grado con que se previene, detecta o reacciona adecuadamente ante al daño (y la destrucción de partes) accidental del ambiente.

8.1.3 La Seguridad como un Factor de Calidad

Los factores de protección y de seguridad se pueden ver como las dos caras de la misma moneda. Si la protección se puede definir como el grado con que se gestiona adecuadamente un daño

accidental, entonces la seguridad se la puede definir como el grado con que se gestiona adecuadamente el daño *malicioso*.

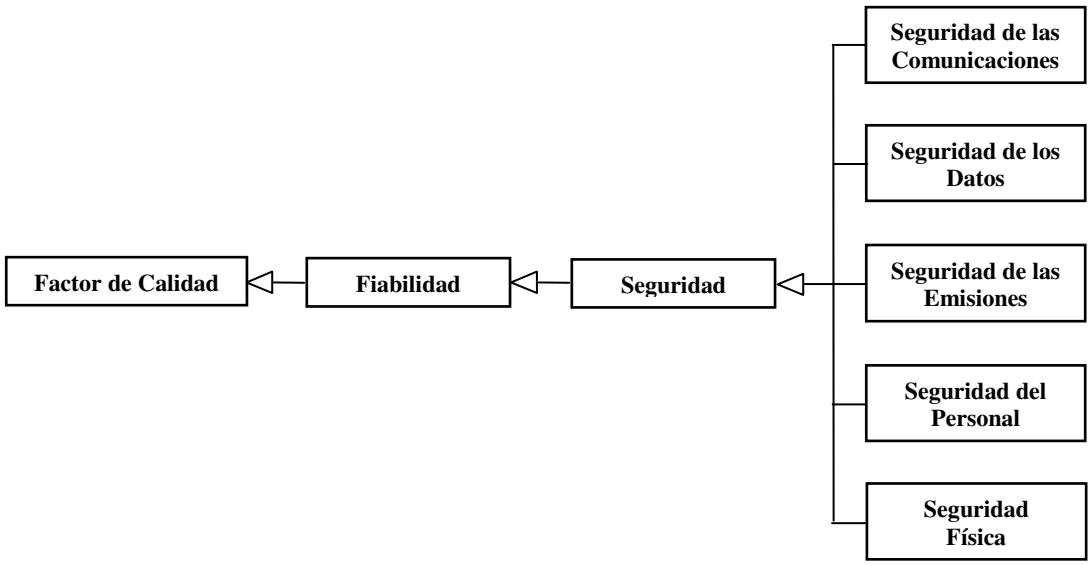


Figura 50. Seguridad como Factor de Calidad.

El modelo de información que se ilustra en la Figura 50 muestra que la seguridad también es una clase de fiablidad y, en consecuencia, una clase de factor de calidad. El modelo de información también muestra que la seguridad tradicionalmente ha estado clasificada en sub-clases tales como seguridad de las comunicaciones, seguridad de los datos, seguridad de las emisiones, seguridad del personal, y seguridad física, especialmente basada (al igual que la protección) en el tipo de activo (valor) que pudiera ser dañado en caso de ocurrir un ataque.

Lo indicado en la Figura 50 se puede definir de la siguiente manera:

- **Seguridad** Grado con el cual se previene, reduce o reacciona apropiadamente ante un daño *malicioso*[25] sobre un activo. Además, es el factor de calidad que indica el grado con el cual los activos son protegidos contra amenazas significativas planteadas por atacantes maliciosos. El factor de calidad de seguridad se puede clasificar dentro de las siguientes sub-clases, las cuales también son factores de calidad:
  - **Seguridad de las comunicaciones** Grado con cual las comunicaciones se encuentran protegidas ante ataques.
  - **Seguridad de datos** Grado con el cual los datos almacenados y manipulados se encuentran protegidos ante ataques

[25] Se puede argumentar que el término "malicioso" es muy fuerte. Pero ¿qué se puede decir respecto de aquéllos que destrozan el sitio Web de una empresa que contamina el medio ambiente? ¿Y qué de aquellos otros que utilizan las computadoras de la empresa para navegar por al Web violando la política de la compañía? El primer ejemplo es un crimen cibernético, y el segundo es un uso de la propiedad desaprobado. En ambos casos, las víctimas podrían estar justificadas al considerar estos actos como maliciosos. Si el término "malicioso" aún suena demasiado duro, se puedo considerar que significa la combinación de desaprobado y daño intencional.

- **Seguridad de emisiones** Grado con el cual los sistemas no emiten radiaciones que son pasibles de ataques.
- **Seguridad del personal** Grado con el cual el personal se encuentra protegido ante ataques.
- **Seguridad física** Grado con el cual los sistemas se encuentran protegidos ante ataques físicos.

La Figura 51 muestra que la seguridad también se puede descomponer (es decir, verla como un conjunto) en muchos diferentes sub-factores de calidad. De hecho, la seguridad históricamente ha sido definida más bien en términos de sus sub-factores más populares (por lo general, disponibilidad, integridad y privacidad) que en términos de sus sub-clases. Debe observarse que la seguridad es un concepto relativamente complejo y que no puede ser adecuadamente tratado sólo en términos de disponibilidad, integridad y privacidad. Desgraciadamente, no existe un estándar ampliamente aceptado que descomponga la seguridad en una clasificación de sus sub-factores de calidad, y estos sub-factores de calidad no poseen definiciones al nivel de estándar. Tal vez este trabajo ayude a estimular el desarrollo de un consenso con relación a la descomposición óptima de la seguridad en sus sub-factores de calidad.

Los sub-factores ilustrados en la Figura 51 se pueden definir de la siguiente manera:

- **Control de acceso** Grado con el cual los sistemas limitan el acceso a sus recursos sólo a sus externos autorizados (por ejemplo, usuarios humanos, programas, procesos, dispositivos, u otros sistemas). Los siguientes son sub-factores de calidad del sub-factor de calidad de control de acceso:
  - **Identificación** Grado con el cual el sistema identifica (es decir, reconoce) a sus externos antes de interactuar con ellos.
  - **Autenticación** Grado con el cual el sistema verifica las identidades pretendidas de sus externos antes de interactuar con ellos. En consecuencia, la autenticación verifica que la identidad pretendida es legítima y que pertenece a quien la pretende.
  - **Autorización** Grado con el cual son otorgados el acceso e impuestos los privilegios de uso a los externos autenticados.
- **Detección de ataque/daño** Grado con el cual son detectados, registrados y notificados los intentos de ataques o su exitosa ocurrencia (o sus daños resultantes).
- **Protección de disponibilidad** Grado con el cual se previene que diferentes tipos de ataques DoS disminuyan la disponibilidad operativa del sistema. Resulta algo diferente del tradicional factor de calidad disponibilidad, el cual tiene que ver con la disponibilidad operativa del sistema cuando no se encuentra bajo ataque.

- **Integridad** Grado con el cual los componentes se encuentran protegidos frente a la corrupción intencional o desaprobada. La integridad incluye:
    - **Integridad de datos** Grado con el cual los componentes de datos (ya sea almacenados, procesados o transmitidos) se encuentran protegidos frente a la corrupción intencional (por ejemplo, vía creación, modificación, borrado o repetición desaprobados).
    - **Integridad del hardware** Grado con el cual los componentes de hardware se encuentran protegidos frente a la corrupción intencional (por ejemplo, vía adición, modificación desaprobados, o robo).
    - **Integridad del personal** Grado con el cual los componenetes humanos se encuentran protegidos de corrupción intencional (por ejemplo, vía soborno o extorsión).
    - **Integridad del software** Grado con el cual los componentes de software se encuentran protegidos de la corrupción intencional (por ejemplo, vía adición, modificación, borrado desaprobados o robo).
        - **Inmunidad** Grado con el cual el sistema protege sus componentes de software frente a la infección causada por programas maliciosos (es decir, *malware –MALicious software*- tal como virus, gusanos, troyanos bombas lógicas, *scripts* maliciosos, *spyware*). Los componentes de software incluyen programas completos, programas parciales, procesos, tareas y *firmware*.
- **No-repudiación** Grado con el cual una parte de una interacción (por ejemplo, mensaje, transacción, transmisión de datos) se encuentra protegida ante una repudiación exitosa (es decir, negación) de cualquier aspecto de la interacción. La no-repudiación comprende información tal como las identidades del emisor y del receptor de la transacción; el momento de envío, el de recepción y las fechas de la transacción; y cualquier dato que fluye dentro de la transacción. Por ello, la no-repudiación asume la integridad de los datos de tal manera que una parte no puede argumentar que los datos asociados estaban corruptos.
- **Protección física** Grado con el cual el sistema se protege a sí mismo y a sus componentes de un ataque físico. El ataque físico puede significar algo tan violento como el uso de una bomba o el secuestro o la extorsión del personal. También puede significar algo tan relativamente menor como el prevenir el robo de una computadora portátil mediante un cable o un candado.
- **Privacidad** Grado con el cual se previene que partes desaprobadas obtengan información sensible. La privacidad incluye los siguientes sub-factores:

- **Anonimato** Grado con el cual se previene que las identidades de los usuarios sean almacenadas o reveladas de manera desaprobada.
- **Confidencialidad** Grado con el cual se previene que información sensible sea revelada a partes desaprobadas (por ejemplo, individuos, programas, dispositivos y otros sistemas).

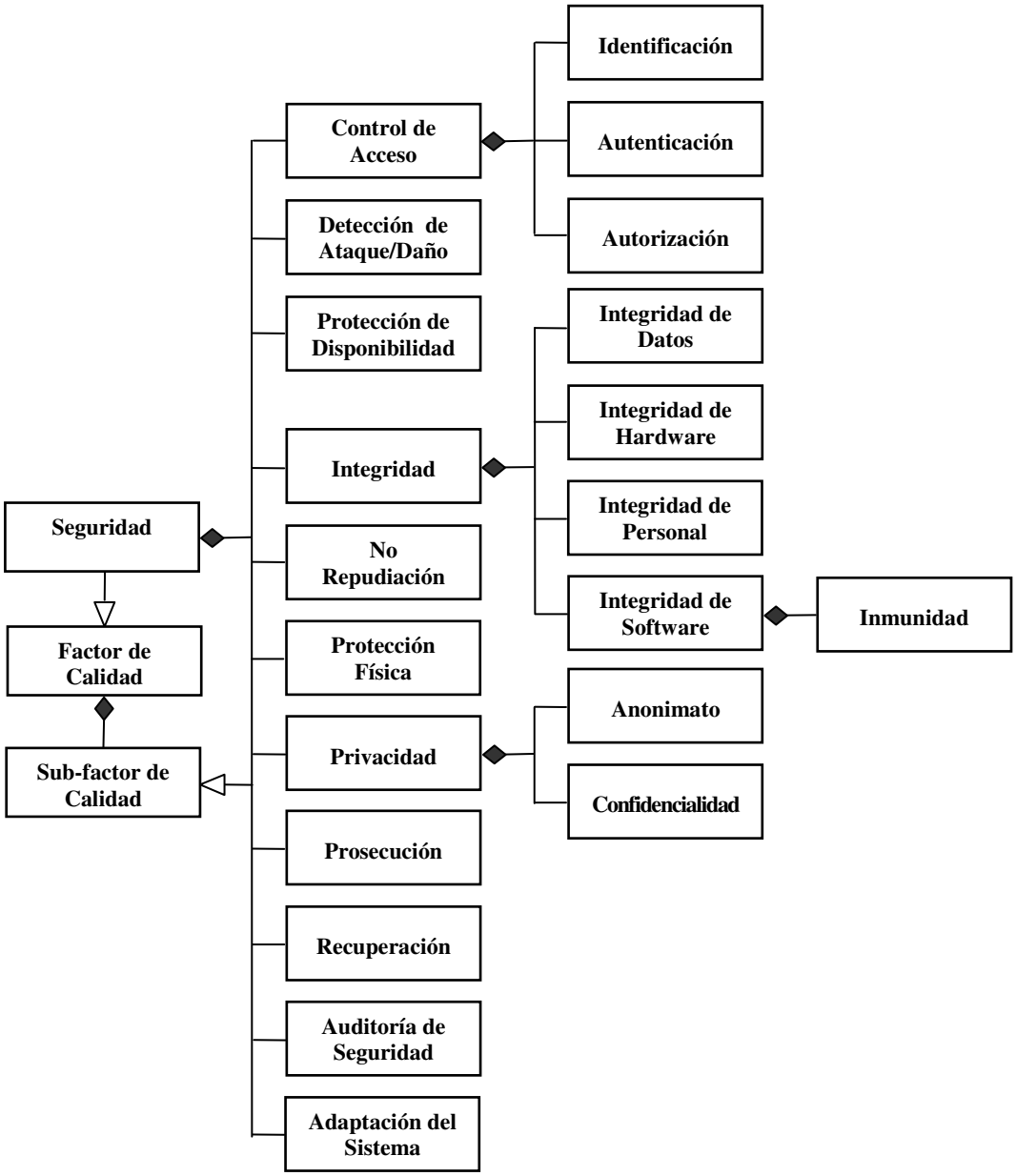


Figure 51: Descomposición de la Seguridad en Sub-Factores de Calidad.

- **Prosecusión** Grado con el cual el sistema soporta la acción de los atacantes.
- **Recuperación** Grado con el cual el sistema de recupera luego de un ataque exitoso.
- **Auditoría de seguridad** Grado con el cual el personal de seguridad es capaz de auditar el estado y uso de los mecanismos de seguridad mediante el análisis de los eventos rela-

cionados con la seguridad.

- **Adaptación del sistema** Grado con el cual el sistema aprende de los ataques a los fines de adaptar sus contramedidas de seguridad para protegerse a sí mismo de ataques similares en el futuro.

8.1.4 La Supervivencia como un Factor de Calidad

La supervivencia tiene que ver con los servicios de misión crítica esenciales que deben continuar siendo provistos a pesar de un daño accidental o malicioso [Ellison 99a, Ellison 03, Knight 00a, Knight 03, Lipson 99]. En consecuencia, en cierto sentido, la supervivencia se puede ver tanto como una unión de protección y seguridad (daño accidental y malicioso) como un subconjunto de ellos (sólo lo que se interesa en daños de los servicios esenciales).

La supervivencia por lo general no está subdividida en clases de menor nivel de acuerdo al tipo de activo protegido debido a que sólo tiene que ver con los servicios esenciales; en consecuencia, sólo existe un tipo de activo que proteger. Sin embargo, al igual que la seguridad, la supervivencia general se descompone en sub-factores de cualidad. Específicamente, la supervivencia comprende la prevención, detección, y reacción, tal como lo ilustra el diagrama de clases de la Figura 52[26].

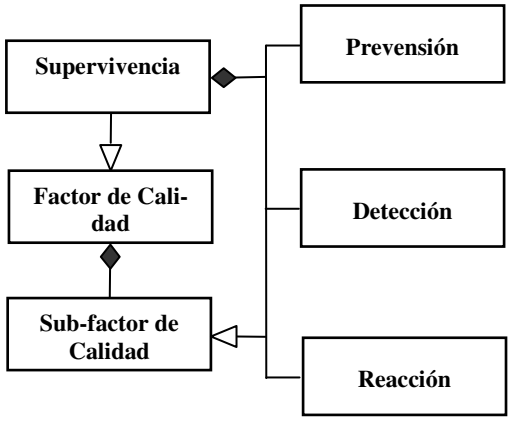


Figura 52. Descomposición de la Supervivencia en Sub-factores de Calidad.

La supervivencia y sus sub-factores se pueden definir de la siguiente manera:

- **Capacidad de supervivencia** es el grado con el cual los servicios esenciales continúan siendo provistos a pesar de daños *maliciosos* o *accidentales*. Tal como lo establecieran H. Lipson y D. Fisher [Lipson 99], "A fin de cuentas, es la misión (del sistema) lo que debe sobrevivir, no algún componente particular del sistema o incluso el sistema mismo". Los siguientes son los sub-factores de calidad del sub-factor capacidad de supervivencia:

[26] El uso de un único yoke conectado tanto a un diamante negro (agregación) como a una flecha blanca (herencia) no es exactamente válido en UML, pero dibujar dos yokes superpuestos podría introducir significativa confusión en el diagrama. Por ejemplo, los sub-factores de calidad (flecha de herencia) prevención, detección y reacción son componentes (diamante de agregación) de la supervivencia.

- **Prevención** (también conocido como resistencia) es el grado con el cual son resistidos los peligros y las amenazas de tal manera que los servicios esenciales continúan siendo provistos tanto durante como después de los accidentes y ataques. La prevención incluye tanto la eliminación de tales peligros y amenazas como los pasos que se toman para minimizar los resultados negativos que un accidente o un ataque exitosos podrían causar.
- **Detección** (también conocido como reconocimiento) es el grado con el cual son reconocidos los accidentes y ataques relevantes (o el daño por ellos causado) mientras están sucediendo, de tal manera que el sistema pueda reaccionar de manera acorde para mantener sus servicios esenciales. Generalmente, también comprende la registración de los ataques a fin de que existan evidencias con las cuales enjuiciar a los atacantes. La detección también puede incluir el reconocimiento de condiciones o eventos que preceden a los accidentes (por ejemplo, la falla inminente de hardware) o la detección de atacantes recolectando información durante los sondeos previos a los ataques.
- **Reacción** (también conocido como recuperación) es el grado con el cual un sistema responde (es decir, se recupera) luego de un accidente o ataque. Esta reacción incluye el establecimiento de una metodología de recuperación basada en prioridades de tal manera que cualquier servicio esencial que pudiera haber sido perdido o degradado sea recuperado antes que cualquier otro servicio no-esencial que se pudiera haber perdido o terminado. Si bien algunos autores [Mead 03] han puesto de manifiesto que la recuperación diferencia la supervivencia de la protección y de la seguridad, la reacción que incluye la recuperación no sólo se debería aplicar a los servicios esenciales sino también al daño que hubiera sufrido cualquier activo. Por ejemplo, la seguridad se encuentra desafortunadamente concentrada en la prevención y se ignora de qué manera los sistemas deberían detectar y reaccionar frente a ataques.

8.1.5 Síntesis

Hasta aquí se ha presentado el concepto de un modelo de calidad, como así también sus componentes y sus interrelaciones, y se ha realizado una introducción a las definiciones de los factores de calidad de protección, seguridad y supervivencia. Con sus figuras y definiciones, esta sección nos permite arribar a las siguientes conclusiones:

- *La calidad de un sistema basado fundamentalmente en software no es un concepto simple. Para que resulte de utilidad, se lo debe descomponer en factores y sub-factores de calidad, lo que nos permite hablar de diferentes aspectos específicos de calidad.*

- *Existen muy diferentes clases de factores de calidad, tanto orientados al desarrollo como al uso. Estos factores de calidad se vuelven parte de los fundamentos para la organización, identificación y análisis de los requerimientos de calidad.*
- *Protección, seguridad y supervivencia son factores de calidad que se pueden descomponer en sub-factores de calidad estándares que captan los diferentes aspectos fundamentales de qué es lo que éstos significan en lo que hace a que un sistema basado fundamentalmente en software esté protegido, sea seguro y posea capacidad de supervivencia.*
- *Si bien la protección, la seguridad y la supervivencia son factores de calidad diferentes con diferentes sub-factores, en realidad se encuentran relacionados entre sí de tal manera que entre los tres representan el grado con el cual se prevenga, se proteja o se reaccione ante un daño (ya sea accidental, malicioso o ambos).*
- *La protección, la seguridad y la supervivencia son diferentes (si bien muy similares) factores de calidad. Un sistema puede estar protegido y aún así no ser ni seguro ni poseer capacidad de supervivencia en tanto y en cuanto una persona o una propiedad sufra daños cuando un ataque común con éxito provoque la pérdida de un servicio esencial (por ejemplo, un típico ataque de denegación de servicio). De manera similar, los sistemas pueden ser seguros sin estar protegidos ni poseer capacidad de supervivencia en caso que un accidente provoque la pérdida de un servicio esencial. Finalmente, un sistema puede poseer capacidad de supervivencia sin encontrarse protegido ni ser seguro en tanto y en cuanto accidentes y ataques frecuentes no causen la pérdida de un servicio esencial.*
- *Estos factores y sub-factores de calidad que conforman el modelo de calidad a su vez se los juzga por el criterio de calidad específico para una aplicación y se lo mide mediante métricas de calidad asociadas.*
- *Los criterios de calidad aparecen cuando los factores y sub-factores se vuelven muy específicos y particulares de la aplicación, y se los mide mediante métricas de calidad asociadas. A menudo existen muy variados criterios de calidad que se pueden seleccionar para cualquier factor y sub-factor de calidad, y estos criterios muy a menudo se encuentran parametrizados empleando una plantilla estándar con partes variables.*
- *Las métricas de calidad describen el nivel real o el requerido para algún factor o sub-factor de calidad. Esta sección ha sido presentada a los fines de proporcionar una fundamentación para los requerimientos de ingeniería de protección, seguridad y supervivencia debido a que tales requerimientos son, a menudo, una combinación de criterios y métricas de calidad para los factores de calidad de protección, seguridad y supervivencia.*

8.2 MODELOS DE REQUERIMIENTOS

Los requerimientos apropiados son críticos para la creación de sistemas e basados en software que presenten capacidades de protección, seguridad y supervivencia. Sin embargo, los conceptos relacionados con los objetivos, políticas y componentes arquitectónicos de seguridad a menudo se confunden con sus requerimientos. En esta sección se intenta clarificar las diferencias entre estos conceptos como así también sus relaciones con los modelos de calidad discutidos en la sección 2.1 "Modelos de Calidad".

8.2.1 Modelo de Información para Requerimientos

Un modelo de requerimientos debe documentar las diferentes clases de requerimientos principales (por ejemplo, requerimientos funcionales, de calidad, de datos, de interfase y de restricciones) [Firesmith 02]. *También deberá documentar las relaciones entre los requerimientos y otros conceptos tales como objetivos, políticas y mecanismos arquitectónicos.*

La Figura 53 muestra los requerimientos funcionales que son una clase de requerimiento sin importar cuán críticos sean. Los requerimientos de datos, calidad e interfase como así también de restricciones también están identificados, analizados, especificados y administrados. Asimismo, el diagrama establece que los requerimientos de protección, seguridad y supervivencia son requerimientos de calidad, pero no funcionales, que se traducen en requerimientos arquitectónicos significativos, que tienen un impacto sobre la arquitectura (y el costo de aplicación y la planificación del desarrollo) mucho mayor que la mayoría de los requerimientos funcionales.

Dicha figura está incompleta debido a que no muestra todas las clases de requerimientos de calidad posibles (es decir, aquéllos que especifican las cantidades obligatorias de los otros factores de calidad listados de la taxonomía precedente). Como tal, es una super-simplificación y, en consecuenci,a no constituye un argumento fuerte con respecto a las similitudes entre los requerimientos de protección, seguridad y protección. Cabe destacar que diferentes modelos de calidad y de requerimientos pueden descomponerse en cosas diferentes y, en consecuencia, producir diagramas diferentes.

Si bien los especialistas en requerimientos tradicionalmente no han reconocido ningún elemento de modelo intermedio entre objetivos y requerimientos, las comunidades dedicadas a la protección y la seguridad están muy familiarizadas con las políticas de protección y de seguridad y han comprendido correctamente que estas políticas críticas están por debajo de los objetivos pero por encima de los requerimientos dentro de esta jerarquía. En consecuencia, se incluyen las políticas entre los objetivos y los requerimientos en nuestro modelo de información de requerimientos.

De manera similar, los componentes arquitectónicos como así también las restricciones arquitectónicas se han incluido de manera explícita en la Figura 53 porque muchos especialistas en

requerimientos (como así también especialistas en protección y en seguridad) equivocadamente especifican componentes arquitectónicos (por ejemplo, salvaguardas y contramedidas) como restricciones arquitectónicas en lugar de especificar los verdaderos requerimientos de protección y de seguridad subyacentes. En su lugar, ellos deberían dejar la selección de los componentes arquitectónicos de protección y de seguridad a los equipos encargados de la arquitectura, la protección y la seguridad.

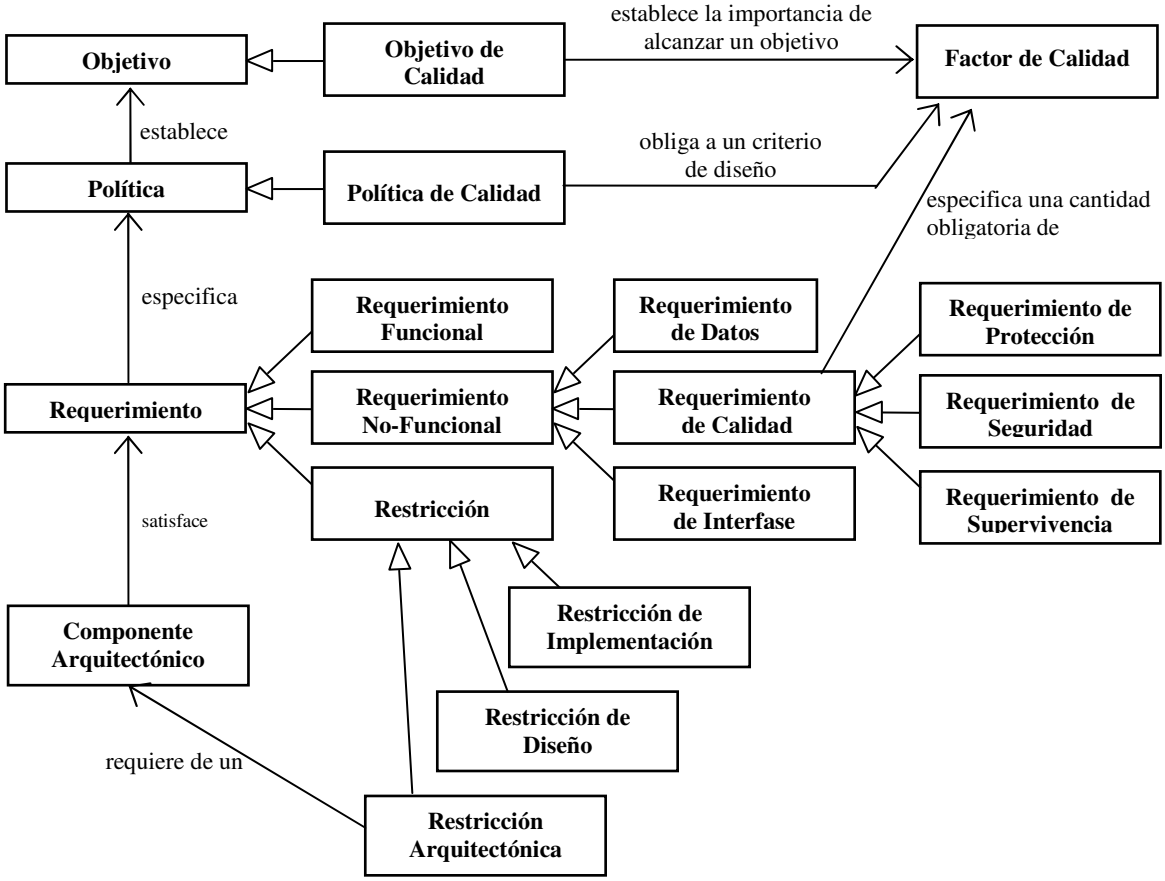


Figura 53. Modelo de Información para Requerimientos

Los conceptos indicados en la Figura 53 se definen de la siguiente manera:

- **Objetivo** es una declaración de la importancia establecida en alcanzar una meta deseada con vista a algún comportamiento, información, característica, interfase o restricción. Se encuentra por encima de una política y no está suficientemente formalizada como para ser verificable.
  - **Objetivo de Calidad** es un objetivo que establece la importancia establecida de alcanzar una meta deseable en vista a algún factor o sub-factor de calidad. (Por ejemplo, "La información sensible debe estar segura" o "La confidencialidad y la integridad de la información sensible deben estar garantizadas").

- **Política** es una decisión estratégica que establece un objetivo deseado.
  - **Política de Calidad** es una política que impone un criterio deseado (o un tipo de criterio) referido a un factor o sub-factor de calidad. (Por ejemplo, "Toda la información acerca de las tarjetas de crédito de los clientes que han sido confiadas a nuestra organización deberá contar con una combinación de medidas de seguridad tecnológica y de procedimiento que, en conjunto, aseguren que todos los tipos de ataques actualmente conocidos serán prevenidos de causar acceso no-autorizado, modificación o robo de esta información".)
- **Requerimiento** es cualquier comportamiento, información, característica o interfase obligatoria, externamente observable, verificable (es decir, testeable) y validable. Debido a que está relacionado con la calidad de un sistema o de sus componentes, el término "requerimiento" se utiliza para hacer referencia a "requerimiento de producto". En consecuencia, no se comtemplan otros tipos de requerimientos (por ejemplo, requerimientos de proceso tales como costos y planificación de desarrollo) que poseen el mismo nivel de abstracción.
  - **Requerimiento funcional** es cualquier requerimiento que especifica un comportamiento obligatorio.
  - **Requerimiento no-funcional** es cualquier requerimiento que especifica un requerimiento no-funcional. Algunos autores equivocadamente consideran equivalentes los requerimientos no-funcionales con los requerimientos de calidad. Esta taxonomía muestra claramente que existen requerimientos no-funcionales que no son requerimientos de calidad. Los tipos de requerimientos no-funcionales incluyen los siguientes:
    - **Requerimiento de datos** es cualquier requerimiento que especifica un aspecto obligatorio de un dato o un tipo de dato.
    - **Requerimiento de calidad** es cualquier requerimiento que especifica una cantidad mínima y obligatoria de un factor de calidad (es decir, característica, atributo). Existen numerosos tipos de requerimientos de calidad, dentro de los que se incluyen:
      - **Requerimiento de protección** es cualquier requerimiento que especifica una cantidad mínima y obligatoria de protección.
      - **Requerimiento de seguridad** es cualquier requerimiento que especifica una cantidad mínima y obligatoria de seguridad.
      - **Requerimiento de supervivencia** es cualquier requerimiento que especifica una cantidad mínima y obligatoria de supervivencia.

        - **Requerimiento de interfase** es cualquier requerimiento que especifica un aspecto obligatorio de una interfase o protocolo externo.
    - **Restricción** es cualquier decisión de ingeniería que ha sido seleccionada como un requerimiento. Existen varios tipos de restricciones, dentro de las que se incluyen:
        - **Restricción arquitectónica** es cualquier decisión arquitectónica que ha sido seleccionada para ser considerada como una restricción obligatoria (es decir, como un requerimiento)
        - **Restricción de diseño** es cualquier decisión que ha sido seleccionada para ser considerada como una restricción obligatoria (es decir, como un requerimiento)
        - **Restricción de implementación** es cualquier decisión que ha sido seleccionada para ser tratada como una restricción obligatoria (es decir, como un requerimiento). Ejemplos que se pueden incluir son incluir la selección de un lenguaje de programación o una manera estándar de utilizar el lenguaje de programación (por ejemplo, un subconjunto "protegido" y/o "seguro" de los constructores del lenguaje).
- **Componente arquitectónico** es una opción arquitectónica que proporciona un medio para satisfacer uno o más requerimientos relacionados. Esta definición *no* incluye sólo la arquitectura de software. Desde un punto de vista del arquitecto del sistema, los componentes arquitectónicos pueden estar implementados por uno o más componentes del sistema (incluyendo hardware, software, datos, personal y documentación). En consecuencia, los mecanismos arquitectónicos pueden incluir los materiales de entrenamiento y los procedimientos operativos tanto para personal interno del sistema como externo. Este punto de vista a nivel de sistemas *reconoce que la protección, la seguridad y la supervivencia no se pueden lograr utilizando solamente hardware, software y datos. Todos los componentes del sistema (incluidas las personas, sus procedimientos, y el entrenamiento que ellas reciben) se deben contribuir a alcanzar un nivel requerido de protección, seguridad y supervivencia.*

8.2.2 Requerimientos de Calidad y Factores de Calidad

En el nivel más alto de abstracción, los objetivos de calidad pueden ser o un factor de calidad o un sub-factor de calidad. Por ejemplo, pueden existir objetivos de calidad concernientes con la protección, la seguridad y la supervivencia. Ejemplos de objetivos de protección podrían ser, "El sistema debe estar protegido" o "El sistema no debe causar daños a sus usuarios". Por debajo de este nivel, pueden existir políticas de calidad más detalladas que establezcan el objetivo de calidad para un criterio obligatorio de calidad específico del sistema (o un tipo de criterio) co-

rrespondiente al factor o sub-factor de calidad asociado. Un ejemplo de política de protección podría ser, "Las partes móviles de la librería automatizada de cintas no deberán dañar a los operadores cuando realizan sus tareas". Finalmente, un requerimiento de calidad específico y verificable viene dado por una combinación de criterio de calidad con un nivel mínimo obligatorio de una métrica de calidad asociada. En consecuencia, la política de protección previa se vuelve un requerimiento de protección cuando se la re-escribe de la siguiente manera: "Al menos el 99,99% de las veces que el operador realiza el caso de uso "remover cinta", las partes móviles de la librería automatizada de cintas no se moverán (pudiendo así causar un daño al operador)."

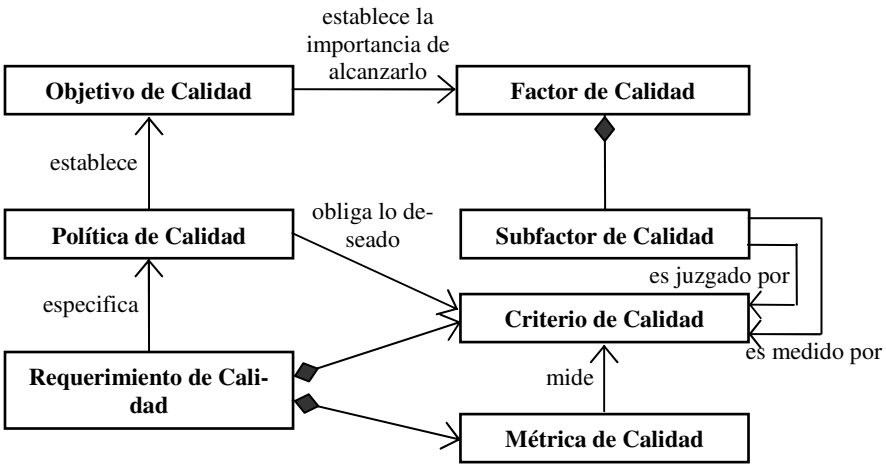


Figura 54. Relaciones entre el Modelo de Requerimientos y el Modelo de Calidad

La Figura 54 relaciona los conceptos de requerimientos con los conceptos de modelo de calidad[27]. Asimismo provee una manera conveniente de dividir el trabajo, en el que los administradores establecen los objetivos y las políticas de calidad, en tanto que el equipo de requerimientos (con la entrada alimentada por los equipos de protección y seguridad) especifica los requerimientos de calidad asociados.

Como se ilustra en la parte superior de la Figura 55, los factores de calidad *caracterizan* diferentes aspectos de la calidad del sistema y los requerimientos de calidad *especifican* diferentes aspectos de la calidad del sistema mediante la especificación de niveles de los factores y subfactores de calidad asociados. En consecuencia, los requerimientos de protección especifican niveles obligatorios de protección, los requerimientos de seguridad especifican niveles obligatorios de seguridad, y los requerimientos de supervivencia especifican niveles obligatorios de supervivencia. La especificación de estos requerimientos se lleva a cabo especificando un criterio de calidad asociado y un valor mínimo para una métrica de calidad. Este modelo de información agrupa las partes de las figuras previas.

[27] El lado izquierdo de la Figura 54 proviene del lado izquierdo de la Figura 53, y el lado derecho de la Figura 54 proviene de la Figura 48.

La Figura 55 muestra de qué manera requerimientos específicos se relacionan con el conjunto de factores de calidad, criterios de calidad y métricas de calidad asociados que se describieron en la sección previa[28]. En el Anexo 1, se presenta un ejemplo de Requerimientos de Calidad.

8.2.3 Síntesis

En las secciones precedentes ha informado sobre las diferentes clases de requerimientos y la manera en que los requerimientos de calidad están relacionados con sus correspondientes factores y sub-factores de calidad. Con estas figuras y definiciones, conduce a las siguientes conclusiones:

- *Los requerimientos de calidad están estrechamente relacionados con los correspondientes componentes del modelo de calidad.*
- *Los requerimientos de calidad se vuelven específicos y verificables al estar basados en una combinación de criterios y métricas de calidad.*

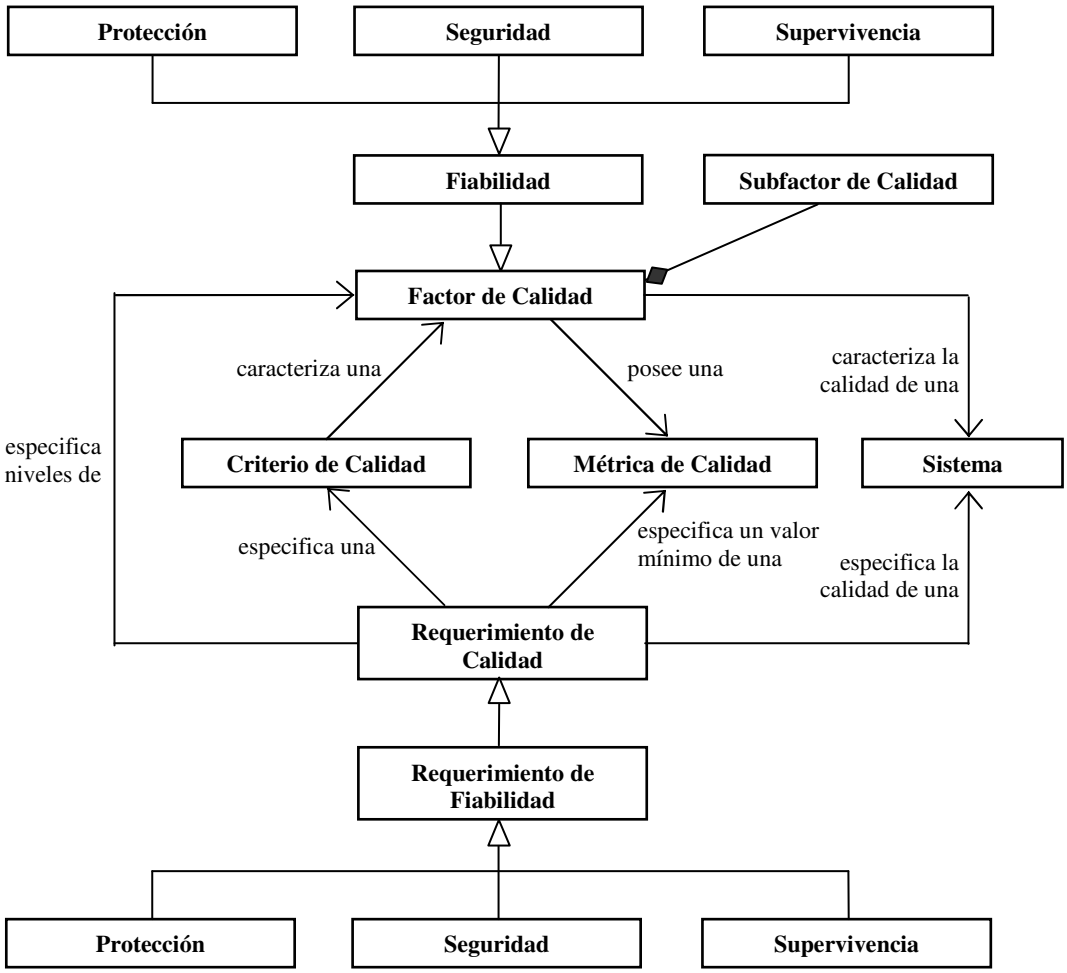


Figura 55. Relaciones entre los Requerimientos de Calidad y el Modelo de Calidad

[28] La Figura 55 está compuesta por partes de las Figuras 48 y 52 y la lista de factores de calidad orientados al uso.

8.3 MODELOS DE INGENIERIA. SIMILITUDES Y DIFERENCIAS

A continuación se documentan los conceptos básicos fundamentales de la ingeniería de protección, seguridad y supervivencia en términos de modelos de información (diagramas de clase UML) y las definiciones asociadas. Construidos a partir de los modelos anteriores, se clarifican las similitudes y las diferencias entre estas tres disciplinas, proporcionando además la base para recomendar su *unificación bajo un nuevo paraguas de ingeniería de la defendibilidad*.

8.3.1 Modelo de Información para la Ingeniería de Protección

Recordemos que los *objetivos de protección* establecen la importancia de alcanzar un nivel deseado del factor de calidad protección, las *políticas de protección* establecen los objetivos de protección mediante la obligatoriedad de los criterios de protección deseados, y los *requerimientos de protección* especifican las políticas de protección mediante la especificación de cantidad obligatorias de protección en términos de criterios de protección específicos con una medida mínima aceptable asociada.

En consecuencia, *los requerimientos de protección requieren la eliminación o la reducción de los riesgos de protección*, los cuales se deben a *peligros*, lo cual proporciona un *framework* organizacional para accidentes similares que *puedan causar daños a activos de valor del* sistema. A su vez, *los requerimientos de protección son satisfechos mediante los componentes arquitectónicos de protección (salvaguardas),* los cuales están destinados a *prevenir o reducir las vulnerabilidades de los valores frente a daños accidentales*. Todos estos conceptos básicos y sus relaciones se ilustran en la Figura 56 la que representa un ***modelo de información para la ingeniería de protección***.

La misma muestra *de qué manera conceptos importantes incluidos en la ingeniería de protección* (por ejemplo, activo, accidente, peligro, riesgo, vulnerabilidad) *se relacionan con términos importantes de la ingeniería de requerimientos* (por ejemplo, objetivo de protección, política y requerimiento)*, con la ingeniería de calidad* (por ejemplo, protección)*, y la arquitectura* (por ejemplo, mecanismo de protección). *También explica y justifica la metodología de análisis de protección vía el análisis de riesgos en términos de vulnerabilidades, peligros, accidentes y activos.*

Los conceptos documentados en dicha figura se pueden definir de la siguiente manera:

- **Accidente** es un evento o serie de eventos no-planeados y no-intencionados (pero no necesariamente no-previstos) que causan un daño a un activo. Los accidentes se clasifican de la siguiente manera:
    - Accidentes contra la salud causan un daño significativo (por ejemplo, enfermedad, lesiones o muerte) a las personas; si bien la ingeniería de protección tiende a enfatizar la protección de las personas contra daños accidentales, su alcance también in-

cluye la propiedad y el ambiente.

- **Accidentes contra la propiedad** causan deterioro o destrucción de las propiedades; el énfasis está puesto sobre las propiedades externas, pero no se deben ignorar los componentes del sistema.
- **Accidentes contra el ambiente** causan un deterioro en el ambiente.

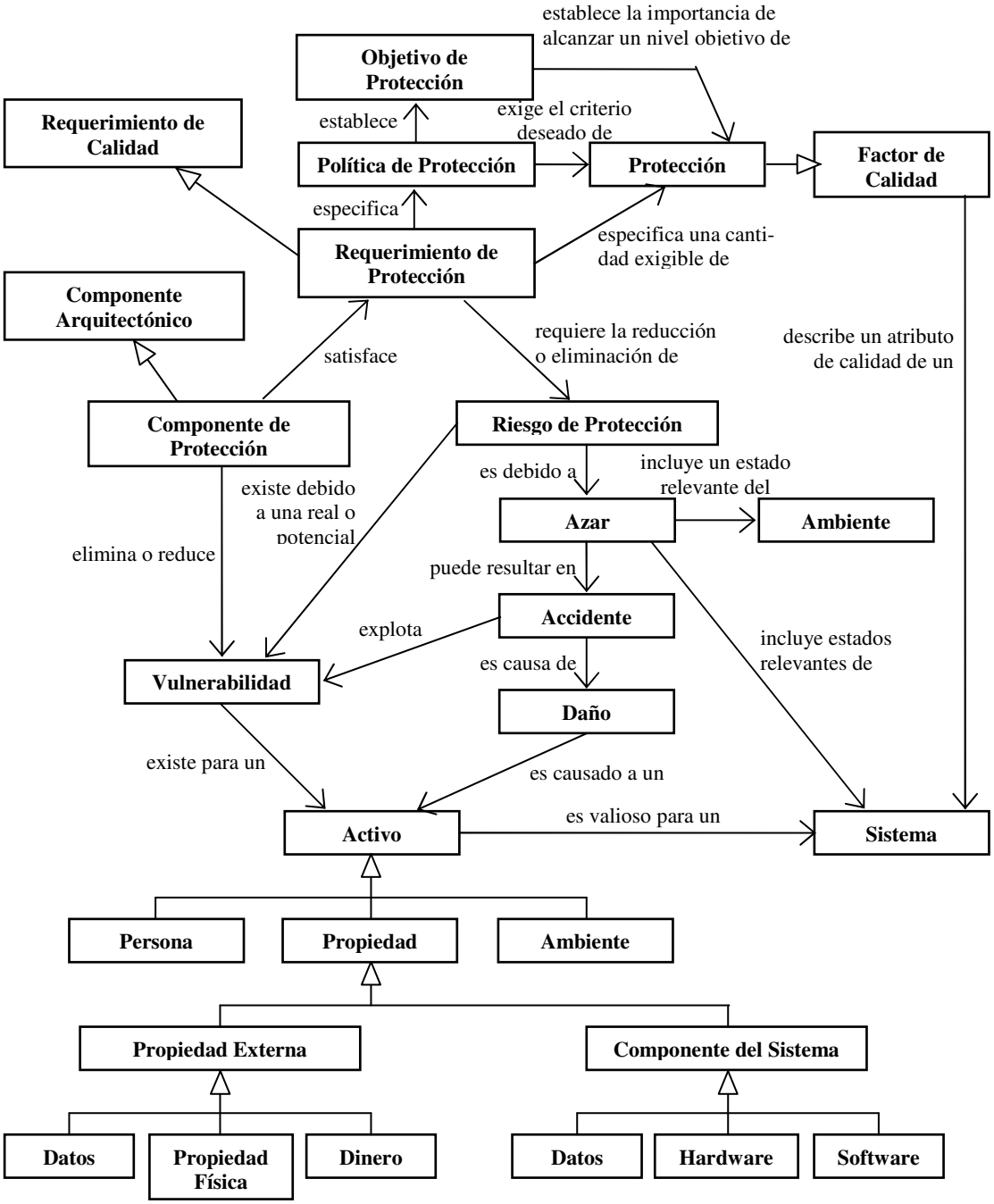


Figura 56. Modelo de Información para la Ingeniería de Protección

- **Activo** (con respecto a la ingeniería de protección) es cualquier valor que debería estar

protegido contra daño accidental. Un activo requiere protección debido a que es un sujeto potencial de sufrir un accidente; puede ser alguno de los siguientes:

- **Personas** (también conocidas como víctimas) seres humanos que son dañados (es decir, desarrollan enfermedades ocupacionales, sufren lesiones, o mueren) como resultado de accidentes, y se las puede clasificar de la siguiente manera:
  - **Víctimas en primera instancia** son víctimas que forman parte del sistema (por ejemplo operadores, administradores, y personal de mantenimiento).
  - **Víctimas en segunda instancia** son víctimas externas al sistema pero que interactúan intencionalmente con el mismo (por ejemplo usuarios y proveedores).
  - **Víctimas en tercera instancia** son víctimas miembros del público general que son espectadores circunstanciales que no se encuentran intencionalmente relacionados con el sistema.
  - **Víctimas en cuarta instancia** son víctimas miembros de generaciones futuras, particularmente víctimas de la radiación, químicos tóxicos y patógenos (por ejemplo fetos no-nacidos, niños cuyos padres no serán capaces de concebir, niños con deformaciones, y generaciones futuras que deberán vivir en ambientes contaminados).
- **Propiedad** es cualquier propiedad de valor que puede ser deteriorada o destruida en caso que ocurra un accidente. Tomando como ejemplo un control de un automóvil, la propiedad podría incluir el auto en sí mismo, y cualquier cosa que pudiera resultar dañada mientras está siendo conducido (tales como otros vehículos, construcciones, señalizaciones, postes telefónicos, semáforos y alumbrado público). La propiedad incluye:
  - **Propiedad externa** es la propiedad personal, comercial y social que existe por fuera del sistema (por ejemplos datos, dinero y propiedad física tales como construcciones e instalaciones).
- **Componente del sistema** es cualquier propiedad que es un componente del sistema (incluidos los componentes de datos, hardware y software)
  - **Ambiente** es el ambiente físico que puede ser deteriorado en caso que ocurra un accidente. Continuando con el ejemplo del control de un automóvil, el ambiente es el ambiente físico que puede ser dañado por accidentes que liberan fluidos (por ejemplo, gasolina, líquido refrigerante o fluido hidráulico) o da inicio a un incendio de pastizales.

- **Daño** (con respecto a la ingeniería de protección) es un deterioro significativo o un impacto negativo asociado con un activo debido a un accidente; el daño debe ser suficien-

temente significativo como para justificar la implementación de medidas correctivas que prevengan este tipo de daño en el futuro[29].

- **Peligro** es una situación que incrementa la probabilidad de uno o más accidentes relacionados. En consecuencia, un peligro consiste en estados peligrosos (es decir, un conjunto de una o más condiciones o estados incompatibles del sistema, posiblemente que incluyan uno o más condiciones dentro del ambiente del sistema) junto con el accidente (tipo) que los mismos pueden causar[30]. Los peligros potenciales se deberían identificar de manera temprana durante la ingeniería de requerimientos o el diseño de la arquitectura, en tanto que los peligros reales pueden ser identificados en los sistemas existentes. Los siguientes son dos ejemplos de estos peligros potenciales y reales, y la identificación de sus diferentes componentes:
  - **Peligro potencial**: las puertas del subterráneo se encuentran cerrándose, cerradas o abriéndose mientras el subterráneo está en movimiento (condiciones peligrosas), lo cual puede hacer que los pasajeros y/o su propiedad (activo) se desprendan (accidente) y sean lastimados, muertos o lesionados (daño).
  - **Peligro real**: los pasajeros y/o su propiedad en las puertas del subterráneo cuando las mismas se están cerrando (condiciones peligrosas) pueden ser aplastados (accidente) y, en consecuencia, lastimados, muertos o lesionados (daño).
- **Protección** es el factor de calidad que indica el grado con el que un daño *accidental* es prevenido, detectado y frente al que se reacciona de manera adecuada.
- **Objetivo de protección** es un objetivo de calidad que establece la importancia de alcanzar el nivel fijado como meta de protección o de uno de sus sub-factores.
- **Política de protección** es la política de calidad que obliga a un criterio de calidad de protección específico del sistema o de uno de sus sub-factores.
- **Mecanismo de protección** (también conocido como salvaguarda o táctica de protección) es un componente arquitectónico (es decir, una decisión estratégica) que ayuda a satisfacer uno o más requerimientos de protección y/o reduce una o más vulnerabilidades de protección.
- **Requerimiento de protección** es un requerimiento de calidad que especifica una cantidad requerida de protección (generalmente un sub-factor de protección) en términos de

[29] El daño es debido a un *accidente* cuando tiene que ver con la ingeniería de *protección*, es debido a un *ataque* cuando tiene que ver con la ingeniería de *seguridad* y puede tener que ser debido tanto a *accidentes* como a *ataques* cuando tiene que ver con la ingeniería de *supervivencia*.

[30] Ejemplos de las principales condiciones internas de peligro incluyen condiciones químicos peligrosos, altos voltajes y maquinaria móvil bajo control automático; un ejemplo más específico podría ser un ascensor que se mueve con sus puertas abiertas, dos estados incompatibles de un elevador. Ejemplos de las principales condiciones externas de peligro incluyen los incendios y los desastres naturales tales como terremotos, inundaciones, huracanes y tornados. La ingeniería de protección está claramente relacionada con la gestión de desastres.

un criterio específico del sistema o un nivel obligatorio mínimos de una métrica de calidad asociada que es necesaria para satisfacer una o más políticas de protección. El criterio específico del sistema también puede abarcar al ambiente del sistema, la infraestructura dentro del cual éste existe, y cualquier suposición acerca del sistema.

- **Riesgo de protección** es el riesgo potencial de daño sobre un activo debido a accidentes. El riesgo de protección se define como la suma (sobre todos los peligros relevantes) de los productos de los dos términos siguientes: (1) el mayor impacto negativo del daño sobre el activo (es decir, su criticidad, severidad o daño) multiplicado por (2) la probabilidad de que peligro derive en un accidente.

  Utilizando la teoría básica de probabilidad condicional, la probabilidad de que un peligro derive en un accidente que cause un daño se puede calcular/estimar como el producto de los siguientes términos: (1) la probabilidad de que el peligro exista, (2) la probabilidad de que también existan otras condiciones (también conocida como latencia), y (3) la probabilidad de que el peligro conduzca a un accidente si se presentan tanto el peligro como las demás condiciones necesarias (también conocida como peligrosidad).

  Los riesgos de protección potenciales se deberían identificar de manera temprana durante la ingeniería de requerimientos o de diseño de la arquitectura, mientras que los riesgos de protección reales se pueden identificar en los sistemas existentes. Los siguientes son dos ejemplos de tales riesgos de protección potenciales y reales con varios de sus componentes identificados:

  - **Riesgo de protección potencial**: a menos que uno o más mecanismos de protección estén instalados (vulnerabilidad potencial) para prevenir que las puertas se abran mientras el subterráneo se encuentre en movimiento (condiciones peligrosas), existe una probabilidad inaceptablemente alta (probabilidad) de que los pasajeros y/o su propiedad (activos) salgan despedidos (accidente) y sufran heridas, daños o mueran (daño).
  - **Riesgo de protección real**: debido a la falta de sensores y el software asociado (vulnerabilidad) para detectar personas y sus propiedades presentes en la puerta del subterráneo cuando las puertas se están cerrando (condiciones peligrosas), existe una probabilidad significativa (probabilidad) de que los pasajeros y/o sus propiedades (activos) resulten aplastados (accidente) y, de esta manera, heridos, dañados o muertos (daño).
- **Vulnerabilidad de protección** es una debilidad del sistema que incrementa la probabilidad de que ocurra un accidente y cause daño. Esta debilidad puede estar en la arquitectura, el diseño, la implementación, la integración, la implantación y la configuración del

sistema (por ejemplo falta de características de protección, falta de mecanismos de alerta, o defectos que podrían causar fallas).

8.3.2 Modelo de Información para la Ingeniería de Seguridad

Recordemos que los *objetivos de seguridad* establecen la importancia de alcanzar un nivel deseado del factor de calidad seguridad, las *políticas de seguridad* establecen los objetivos fijando la obligatoriedad de un criterio de seguridad deseado, y los *requerimientos de seguridad* establecen las políticas de seguridad mediante la especificación de cantidades obligatorias de seguridad en términos de un criterio de seguridad específico con una mínima medida aceptable asociada.

En consecuencia, *los requerimientos de seguridad requieren la eliminación o la reducción de los riesgos de seguridad*, los cuales se deben a *la amenaza de ataque por parte de atacantes con la intención de causar daños a activos de valor del sistema*. A su vez, *los requerimientos de seguridad son satisfechos por los mecanismos arquitectónicos de seguridad (contramedidas)*, los cuales están destinados a *prevenir o reducir las vulnerabilidades de los activos frente a daños maliciosos*. Todos estos conceptos básicos y sus relaciones se ilustran en la Figura 57, la que representa un **modelo de información para la ingeniería de seguridad**.

La misma muestra *de qué manera conceptos importantes de la ingeniería de seguridad* (por ejemplo, activo, ataque, atacante, amenaza, riesgo, política de seguridad, vulnerabilidad) *se relacionan con términos importantes de ingeniería de requerimientos* (por ejemplo, objetivo, política y requerimiento de seguridad). Asimismo, *explica y justifica la metodología de análisis de la seguridad para el análisis de los riesgos en términos de vulnerabilidades, amenazas, ataques, y activos*. Finalmente, sus contenidos y la topología muestran la *clara relación entre ingeniería de protección y de seguridad.*

Los conceptos documentados en dicha figura se pueden definir de la siguiente manera:

- **Activo** (con respecto a la ingeniería de seguridad) es cualquier valor que deberá ser protegido contra daño malicioso, ya que, potencialmente, es pasible de sufrir un ataque; el énfasis tiende a estar en los activos de datos (por ejemplo, integridad y privacidad), pero la también incluye activos de software (por ejemplo, integridad) y servicios (por ejemplo, robo y denegación de servicios), en tanto que la seguridad física se ocupa de proteger personas y otras propiedades, incluidos hardware y equipos.
- **Servicio** es cualquier funcionalidad o capacidad provista por el sistema.

  Ataque (también conocido como brecha de seguridad) es un intento no-autorizado de un atacante que causa daño sobre un activo (es decir, violar la seguridad del sistema, eludir los mecanismos de seguridad); un ataque puede resultar exitoso o no. Debido a su naturaleza maliciosa, la mayoría de los ataques son ciber-crímenes, crímenes (por ejemplo

robo de dinero o de servicios, fraude, espionaje, extorsión, vandalismo, terrorismo, pornografía infantil) llevados a cabo utilizando recursos computacionales. Sin embargo, algunos abusos no-autorizados de los sistemas basados en software son meramente éticos u ofensivos más que criminales.

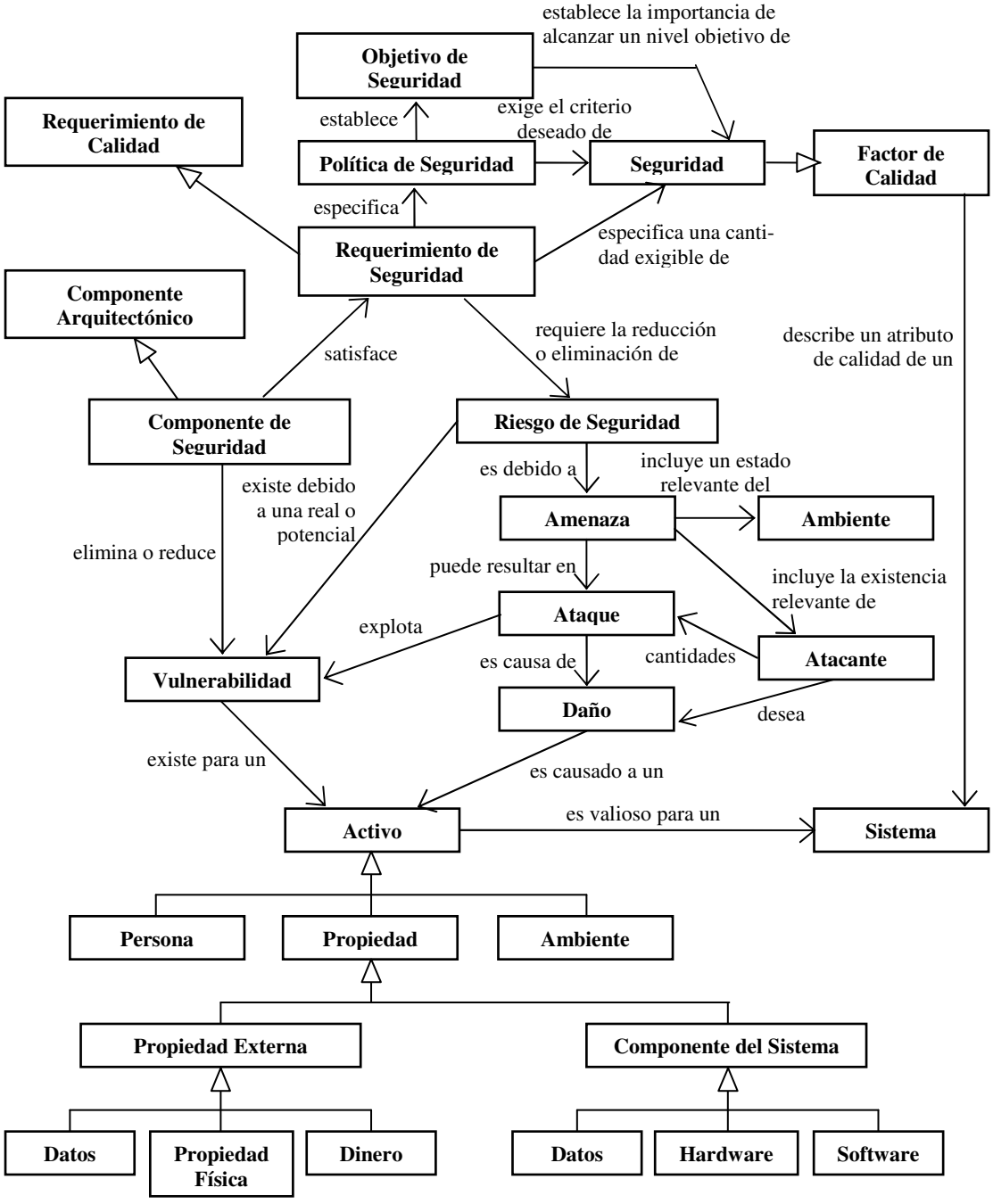


Figura 57. Modelo de Información para la Ingeniería de Seguridad.

- **Atacante** (también conocido como adversario) es un agente (por ejemplo, persona o programa) que causa un ataque debido al deseo de causar daño a un activo.

- **Daño** (con respecto a la ingeniería de seguridad) es un impacto negativo asociado con un activo debido a un ataque[31].
- **Amenaza** es una situación que incrementa la probabilidad de uno o más ataques relacionados; consiste en la existencia de uno o más potenciales atacantes junto con un conjunto de una o más condiciones o estados del sistema que motivan a los atacantes; en consecuencia, la amenaza de robo puede resultar en un robo real (ataque)[32].
- **Seguridad** es el grado con el cual un daño *malicioso* sobre un activo de valor es prevenido, detectado y frente al que se reacciona de manera adecuada; en consecuencia, es el factor de calidad que indica el grado con el cual los activos de valor se encuentran protegidos frente a amenazas significativas planteadas por atacantes maliciosos.
- **Objetivo de seguridad** es un objetivo de calidad que establece la importancia de alcanzar un nivel fijado como meta de seguridad o de uno de sus sub-factores.
- **Política de seguridad** es una política de calidad que obliga a un criterio de calidad de seguridad específico del sistema o de uno de sus sub-factores; estos criterios también pueden comprender el ambiente del sistema, la infraestructura es la cual existe, y cualquier suposición sobre el sistema.
- **Mecanismo de seguridad** (también conocido como contramedida o táctica de seguridad) es un componente arquitectónico (es decir, una decisión estratégica) que ayuda a satisfacer uno o más requerimientos de seguridad y/o reduce una o más vulnerabilidades de seguridad. Se pueden implementar como alguna combinación de componentes de hardware o software, procedimientos manuales, capacitación, etc. También debería observarse que el mismo mecanismo arquitectónico (por ejemplo, redundancia) a menudo se puede utilizar como un mecanismo de protección, seguridad y supervivencia.
- **Requerimientos de seguridad** es un requerimiento de calidad que especifica una cantidad requerida de seguridad (realmente un sub-factor de calidad de seguridad) en términos de un criterio específico del sistema y un nivel mínimo de una métrica de calidad asociada que es necesario alcanzar mediante uno o más políticas de seguridad.
- **Riesgo de seguridad** es el riesgo potencial de daño sobre un activo debido a ataques. El riesgo de seguridad es la suma (aplicada a todas las amenazas relevantes) del impacto negativo del daño sobre un activo (es decir, su criticidad) multiplicado por la probabilidad de ocurrencia del daño[33].

[31] Ver referencia 1 de esta sección.

[32] En mucha bibliografía existente, los términos altamente relacionados pero diferentes de "ataque" y "amenaza" muchas veces son confusos por el hecho de ser utilizados como sinónimos.

[33] Utilizando la teoría básica de probabilidad condicional, la probabilidad de que un daño derive de un ataque se puede calcular/estimar como el producto de los siguientes términos: (1) la probabilidad de que la amenaza de ataque exista, (2) la probabilidad de que otras condiciones (por ejemplo, vulnerabilidades) también existan, y (3) la probabi-

- **Vulnerabilidad de la seguridad** es cualquier debilidad dentro del sistema que incrementa la probabilidad de un ataque exitoso (es decir, uno que cause daño) ocurra. No está restringida sólo a vulnerabilidades debidas a problemas de programación sino que también incluye vulnerabilidades en la arquitectura o diseño del sistema, de qué manera está instalado y configurado el sistema, de qué manera están entrenados los usuarios, etc. Las vulnerabilidades de un sistema pueden abarcan sus componentes de datos, de hardware, de software, de roles humanos (es decir, personal), y de documentos.

8.3.3 Modelo de Información para la Ingeniería de Supervivencia

Recordemos que los *objetivos de supervivencia* establecen la importancia de alcanzar un nivel deseado del factor de calidad supervivencia, las *políticas de supervivencia* establecen objetivos fijando la obligatoriedad de un criterio de supervivencia deseado, y los *requerimientos de supervivencia* especifican las políticas de supervivencia mediante la fijación de cantidades obligatorias de supervivencia las que se establecen vía criterios de supervivencia específicos asociados con una medida mínima aceptable.

En consecuencia, *los requerimientos de seguridad exigen la eliminación o reducción de los riesgos de supervivencia*, los cuales se deben *tanto al peligro de accidentes y a la amenaza de ataques por parte de atacantes que podrían causar daños a los activos de valor del sistema*. A su vez, *los requerimientos de supervivencia se satisfacen mediante los mecanismos arquitectónicos, los cuales están destinados a prevenir o reducir las vulnerabilidades de los activos frente a accidentes y ataques*. Todos estos conceptos básicos y sus relaciones se ilustran en la Figura 58, la que representa un **modelo de información para la ingeniería de supervivencia**.

La similitud del contenido y la topología de las Figuras 56, 57 y 58 muestran *la clara y estrecha relación entre la ingeniería de supervivencia, seguridad y protección*. La Figura 58 demás, justifica la metodología de análisis de supervivencia que emplea el análisis de riesgos en términos de vulnerabilidades, amenazas, peligros y activos. Asimismo, muestra *de qué manera la supervivencia está restringida a los servicios esenciales en lugar de otros tipos de activos*.

Los conceptos adicionales de la Figura 58 no han sido definidos en las sub-secciones previas incluyen lo siguiente:

- **Servicio esencial** es cualquier servicio de misión crítica que debe continuar siendo provisto a pesar de cualquier accidente o ataque.
- **Supervivencia** es el grado con el cual los servicios esenciales de misión crítica continúan siendo provistos a pesar de haber sufrido daños accidentales o maliciosos.

lidad de que la amenaza conduzca a un ataque exitoso en caso que se presenten tanto la amenaza como las demás condiciones necesarias.

- **Objetivo de supervivencia** es un objetivo de calidad que establece la importancia de alcanzar un nivel deseado de supervivencia o de uno sus sub-factores.
- **Política de supervivencia** es la política de calidad que establece como obligatorio un criterio de calidad específico del sistema respecto a la supervivencia o a algunos de sus sub-factores.

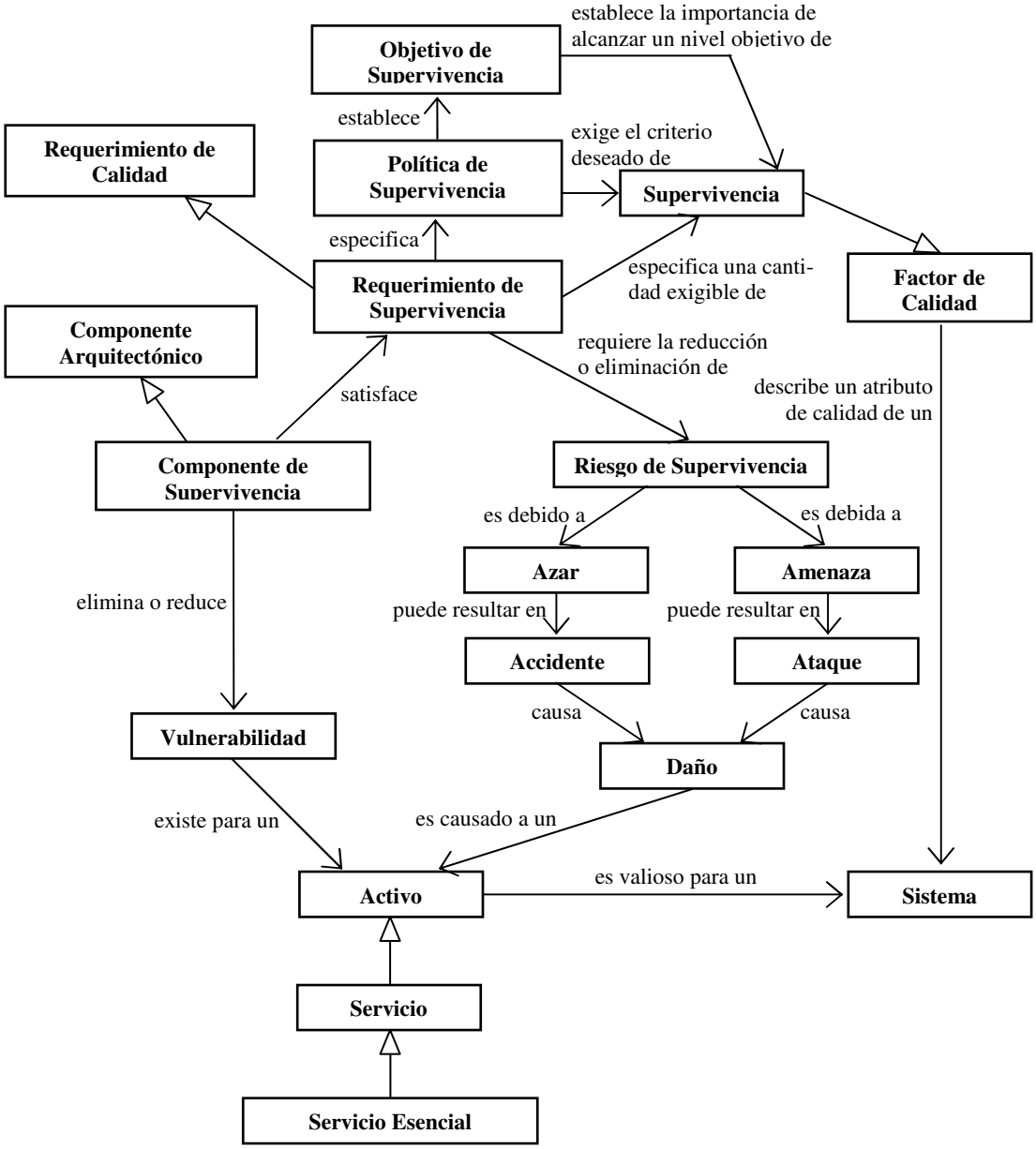


Figura 58. Modelo de Información para la Ingeniería de Supervivencia.

- **Mecanismo de supervivencia** es un mecanismo arquitectónico (es decir, una decisión estratégica) que ayuda a satisfacer uno o más requerimientos de supervivencia y/o reducir una o más vulnerabilidades de supervivencia.
- **Requerimiento de supervivencia** es un requerimiento de calidad que especifica la can-

tidad exigida de supervivencia en términos de criterios específicos del sistema y a un nivel mínimo de la métrica de calidad asociada que es necesario alcanzar en una o más políticas de supervivencia. Los requerimientos de supervivencia generalmente exigen de la identificación de los servicios esenciales de misión crítica (posiblemente como una función del estado del sistema y del tiempo) que se deben proporcionar sin interrupción, la identificación de modos de operación degradados aceptables, la priorización de los servicios alternativos restantes, y el establecimiento del tiempo requerido para lograr que el servicio quede restablecido.

- **Riesgo de supervivencia** es el riesgo potencial de dañar un activo debido a la suma del impacto negativo (considerando todos los peligros y amenazas) del daño sufrido por el activo (es decir, su criticidad) multiplicado por la probabilidad de que ocurra el daño.
- **Vulnerabilidad de supervivencia** es una debilidad del sistema que incrementa la probabilidad que ocurra un accidente o un ataque exitoso y que se detenga la provisión de un servicio esencial.

8.3.4 Síntesis referida a los modelos de ingeniería

Hasta aquí se han documentado los conceptos fundamentales de la ingeniería de protección, seguridad y supervivencia. Con sus figuras y definiciones, conduce a las siguientes definiciones:

- *Los modelos de información de la ingeniería de protección, seguridad y supervivencia son marcadamente similares tanto en contenido como en topología.*
- *Debido a esta consistencia, los requerimientos de protección, seguridad y supervivencia se pueden elicitar y analizar en términos de una metodología basada en activos y orientadas al riesgo que tenga en cuenta la cantidad de peligros y amenazas asociadas con estos activos que deben ser protegidos.*
- *La ingeniería de supervivencia se aproxima (pero no del todo) a la combinación de la ingeniería de protección y de seguridad. La principal diferencia radica en el tipo de activo a ser protegido (en este caso, los servicios críticos que resulta necesario mantener).*

8.3.5 Similitudes y Diferencias

Como se mostró anteriormente, las ingenierías en protección, seguridad y supervivencia son muy similares. A continuación se resumen tanto estas similitudes como las diferencias, proporcionando una base para las recomendaciones específicas que se hacen en la Sección 9.3.

8.3.5.1 Similitudes

Como se ilustró en las Figuras 56, 57 y 58, los modelos de información para protección, seguridad y supervivencia poseen muchas similitudes en contenido y topología; a continuación, se

discuten estas similitudes con más detalle.

Similitudes comunes a todos los requerimientos

Como se destacó en la Figura 53, todos los requerimientos de calidad se pueden organizar dentro de la siguiente cadena jerárquica:

1. Objetivos que guían políticas
2. Políticas que guían requerimientos
3. Requerimientos que guían componentes arquitectónicos
4. Componentes arquitectónicos satisfacen requerimientos

Esta jerarquía se aplica a todos los factores de calidad y no poseen ningún sentido especial para la ingeniería de protección, seguridad y supervivencia.

Similitudes comunes a todos los requerimientos de calidad

Como se destacó en la Figura 55, todos los requerimientos de calidad están relacionados con los factores de calidad. *Las relaciones entre los requerimientos y los factores de calidad tampoco significan nada especial para la protección, seguridad y supervivencia.*

Similitudes específicas

En tanto que las similitudes previas fueron generales, las siguientes son específicas para la ingeniería de protección, seguridad y supervivencia. En las Figuras 56, 57 y 58, estas similitudes incluyen lo siguiente:

Las tres disciplinas:

- *Requieren de la prevención o reducción de los riesgos asociados con peligros y/o amenazas.*
- *Requieren del reconocimiento y respuesta a los accidentes y ataques asociados.*
- *Existen para prevenir, detectar o reaccionar frente al daño que puede sufrir algún activo.*
- *Enfrentan potenciales vulnerabilidades de activos pasibles de dañar.*

8.3.5.2 Diferencias

A continuación se discuten las diferencias esenciales y accesorias entre los modelos de información de las ingenierías de protección, seguridad y supervivencia.

Sub-clases y sub-factores del factor calidad

Protección, seguridad y supervivencia son factores de calidad que difieren en el hecho que, en la actualidad, presentan sub-clases y sub-factores de calidad algo diferentes:

- **Diferentes sub-clases**: *protección y seguridad poseen diferentes sub-clases, en tanto que*

la supervivencia no parece poseer sub-clases. Estas sub-clases de protección y seguridad están esencialmente basadas en los tipos de activos sobre los que pueden ocurrir daños.

- **Sub-clases de protección**
  - Protección de la salud
  - Protección de la propiedad
  - Protección del medio ambiente
- **Sub-clases de seguridad**
  - Seguridad de las comunicaciones
  - Seguridad de los datos
  - Seguridad de las emisiones
  - Seguridad del personal
  - Seguridad física

- **Diferentes sub-factores**: *seguridad y supervivencia poseen diferentes sub-factores, en tanto que la protección no parece poseer sub-factores. Los sub-factores de seguridad y supervivencia no parece estar relacionados estrechamente*.
  - **Sub-factores de seguridad**
    - Control de acceso
      - Identificación
      - Autenticación
      - Autorización
    - Detección de ataque/daño
    - Protección de disponibilidad
    - Integridad
      - Integridad de los datos
      - Integridad del hardware
      - Integridad del personal
      - Integridad del software
      - Inmunidad
    - No-repudiación
    - Protección física
    - Privacidad
      - Anonimato
      - Confidencialidad
    - Auditoría de la seguridad
    - Adaptabilidad del sistema

- **Sub-factores de supervivencia**
  - Prevención
  - Detección
  - Reacción

Aparte del hecho que estas disciplinas han evolucionado de forma significativamente independiente una de otra y que todas ellas están siendo guiadas por los componentes arquitectónicos utilizados para satisfacerlas, no parecen existir razones suficientes por las que las tres descomposiciones anteriores debieran ser tan diferentes. Dado que prevenir es mejor que curar, resulta claro que la prevención ha sido fuertemente enfatizada en las tres disciplinas, mientras que la detección y especialmente la capacidad de respuesta han sido particularmente sub-enfatizadas. *En consecuencia, la mayoría de los sub-factores de seguridad se pueden colocar dentro del marco de la prevención.*

Activos

Protección, seguridad y supervivencia tienden a enfatizar la protección de las diferentes clases de activos frente a diferentes clases de daño.

- **Protección** *enfatiza la protección de las personas frente al daño, si bien también puede y debería proteger la propiedad y el ambiente*.
- **Seguridad** *enfatiza la protección de la propiedad* (datos) *y servicios* (por ejemplo, denegación de servicios) *del daño, si bien también puede y debería proteger personas (*por ejemplo, protección física*) y otras clases de propiedad* (por ejemplo, robo de hardware, posibilidad de sabotaje, e integridad del software frente a virus).
- **Supervivencia** *actualmente se encuentra restringida a la protección de los servicios esenciales frente a daños*[34].

Accidentes y Ataques

Debido a la separación del alcance de la protección y la seguridad en daños accidentales y maliciosos, éstos tienen que ver con diferentes (pero relacionados) tipos de accidentes, y la supervivencia tiene que ver con ambas clases de incidentes.

- **Requerimientos de protección** protegen activos frente al daño causado por accidentes.
- **Requerimientos de seguridad** protegen activos frente al daño causado por ataques.
- **Requerimientos de supervivencia** protegen activos (servicios esenciales) frente al da-

[34] No obstante, el significado de "esencial" muchas veces no es ni absoluto ni constante. Pueden existir múltiples conjuntos de servicios válidos con una secuencia de prioridad que se pueden cambiar debido a modificaciones en las circunstancias externas. Además, la protección y la seguridad de otros activos pueden afectar directamente a la capacidad del sistema de proveer servicios esenciales y, en consecuencia, satisfacer su misión principal

ño debido tanto a accidentes como a ataques.

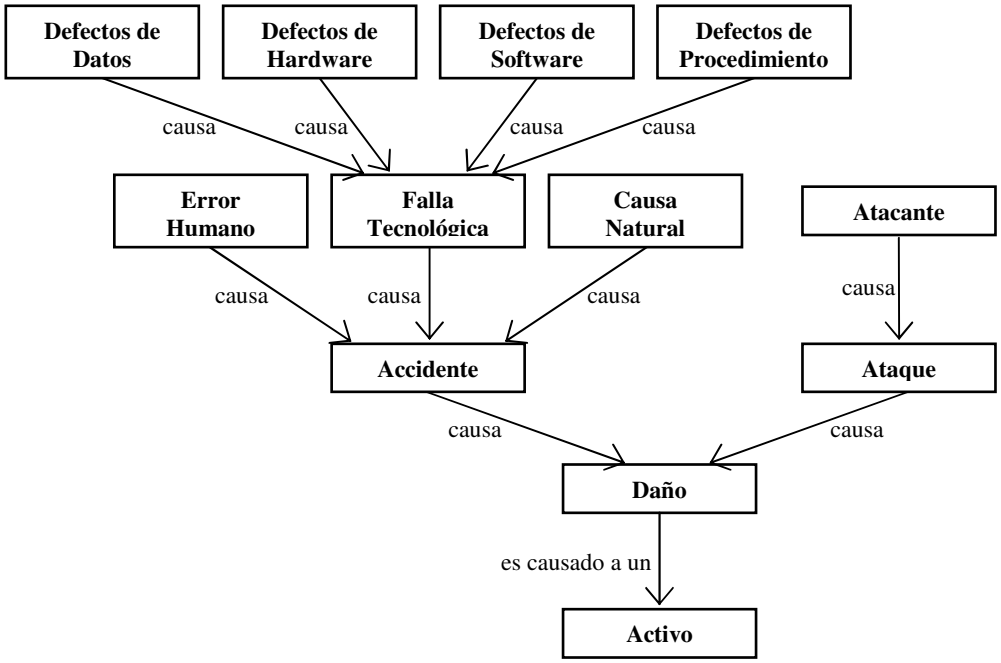


Figura 59. Accidentes vs. Ataques.

Como se ilustra en la Figura 59, *la fuente de daños relevante difiere con respecto a las diferentes clases de requerimientos*:

- **Ingeniería de protección** *atiende a accidentes que son debidos a errores humanos* (por ejemplo, error de operador o de usuario), *fallas tecnológicas* (por ejemplo, fallas resultantes de defectos), *y causas naturales tales como científicas* (por ejemplo, la física de la electricidad, la química de los explosivos)*, ingenieriles* (por ejemplo, "desgaste debido al uso"), *o casos de fuerza mayor* (por ejemplo, desastres naturales).
- **Ingeniería de seguridad** *atiende a ataques que son montados por atacantes* (por ejemplo, personas, organizaciones, herramientas de software).
- **Ingeniería de supervivencia** *atiende a daños provocados tanto por accidentes como por ataques*.

Peligros vs. Amenazas

Debido a la separación del alcance de la protección y de la seguridad en daños accidentales y maliciosos, la seguridad y la protección tiene que ver con diferentes tipos de peligrosidades, y la supervivencia tiene que ver con ambos.

La Figura 60 ilustra las siguientes relaciones:

- **Requerimientos de protección** *protegen los activos frente a daños debidos a peligros*.
- **Requerimientos de seguridad** *protegen los activos frente a daños debidos a amenazas*.

- **Requerimientos de supervivencia** *protegen los activos (servicios esenciales) frente a daños debidos tanto a peligros como a amenazas.*

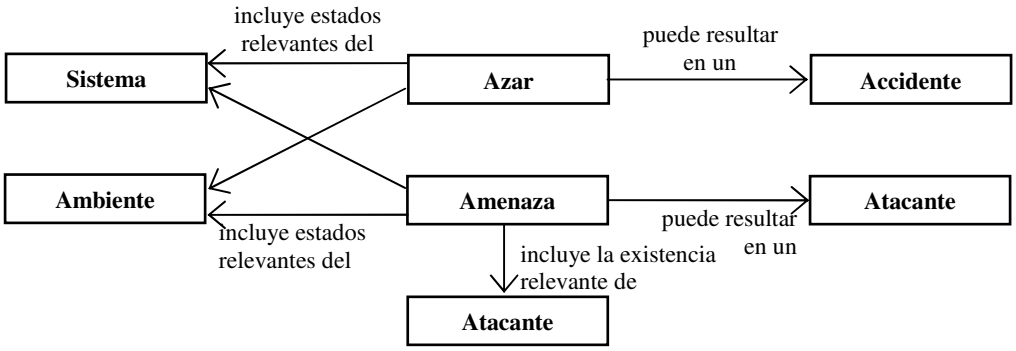


Figura 60. Peligros vs. Amenazas.

8.3.6 Síntesis

Como se puede observar a partir de los modelos básicos documentados en el trabajo de referencia, la protección, la seguridad y la supervivencia se encuentran muy estrechamente relacionadas. Los tres factores de calidad se basan en la protección ante daños de los activos de valor, si bien históricamente cada uno de ellos ha puesto el énfasis de clases diferentes de activos. Los tres incluyen el análisis y la gestión de riesgos basados en potenciales consecuencias negativas y y la probabilidad de su ocurrencia. Asímismo, los tres comprenden requerimientos para prevenir, reducir u ordenar la respuesta o adaptación adecuada por parte del sistema ante la ocurrencia de estos riesgos. Estos requerimientos por lo general son satisfechos por mecanismos arquitectónicos adecuados (por ejemplo, salvaguardas, prácticas y contramedidas de seguridad, mecanismos de supervivencia) que hacen frente a vulnerabilidades potenciales y reales.

Los requerimientos de protección, seguridad y supervivencia se pueden definir como requerimientos de calidad para los factores de calidad de protección, seguridad y supervivencia, respectivamente. Como tales, pueden ser organizados como sub-factores de calidad asociados. También se los puede descomponer en dos partes: un criterio de calidad (una descripción específica que proporciona evidencia ya se a favor o en contra de la existencia de un factor o de un sub-factor de calidad específicos) y una métrica de calidad (un nivel mínimo basados en una escala de medida asociada). En los tres casos, el criterio de calidad generalmente comprende el activo que está siendo protegido y el peligro (por ejemplo, azar o amenaza) del cual está siendo protegido.

El trabajo de referencia provee un fundamente relativamente formal para la ingeniería de la protección, la seguridad y la supervivencia mediante la definición de un modelo de información para:

- Un modelo de calidad y sus partes componentes
- Requerimientos con el énfasis puestos en los requerimientos de calidad para la protec-

ción, la seguridad y la supervivencia

- La protección, la seguridad y la supervivencia.

En el mismo se han identificado los conceptos básicos esenciales subyacentes en las ingenierías de protección, seguridad y supervivencia, proporcionando rigurosas definiciones, ilustrando la relaciones más importantes existentes entre ellos, y proveyendo un serie de recomendaciones en base a lo común existentes en la protección, la seguridad y la supervivencia. De esta manera, resulta posible clarificar y estandarizar la terminología asociada con estos tres campos y así, simplificar y mejorar la ingeniería de sistemas complejos.

SECCION 9: CONCLUSIONES

9.1 CONCLUSIONES GENERALES

Esta Tesis está orientada a relevar y analizar nuevos modelos emergentes en el campo de la arquitectura de la seguridad, a fin de alentar la construcción de sistemas basados en software cuya arquitectura se encuentre contenida dentro de un framework que considera a la "seguridad" como un requerimiento de calidad con identidad propia. Su principal audiencia son los líderes de proyecto, los arquitectos de sistemas, y los ingenieros de software que deben atender requerimientos asociados con la construcción de sistemas de información robustos, o defendibles, como se propone en la misma.

La principal motivación que guía este trabajo fue la búsqueda de respuestas a preguntas tales como:

- ¿Qué es lo que constituye una arquitectura de sistema, y en particular, para un sistema de información que garantice la satisfacción de la misión de la organización donde provee su servicio?
- ¿De qué manera la seguridad afecta los demás requerimientos esenciales de la arquitectura de sistema?
- ¿De qué manera tomar decisiones de cara a la casi demoledora oferta de productos relacionados con la seguridad?
- ¿Cómo llevar adelante la integración de las diferentes opciones?
- ¿Cuáles son los principales inconvenientes que se presentan durante este proceso?
- ¿De qué manera se pueden justificar los costos asociados con la inclusión de requerimientos de seguridad en un sistema de información?

La Ingeniería de Software, que en los últimos 10 años ha venido creciendo de una manera respetable, está dejando en claro que cada vez más profesionales del software comienzan a cumplir roles de arquitectos frente a la necesidad de reconocer que los sistemas de información a escala organizacional son cada vez más complejos, costosos y distribuidos, y que demandan de la atención de requerimientos de calidad tales con *disponibilidad*, *escalabilidad*, *robustez*, e *interoperabilidad*. Simultáneamente, la *seguridad* está ocupando un lugar central en el razonamiento de los líderes de proyecto; existe una comunidad significativamente diversa de profesionales especializados en seguridad de sistemas de información que ofrecen sus valiosos, y a veces, costosos servicios a estos arquitectos de sistema; estos últimos tienen claras exigencias de implementar la política de seguridad de la organi-

zación, y muchos de ellos requieren, por cierto, de una guía de cómo hacerlo.

El proceso de desarrollo de software convierte requerimientos en sistemas, productos y servicios. Y la arquitectura de software ha emergido como un importante principio organizador a nivel de *framework*. Las organizaciones han comenzado a reconocer el valor que posee contar con las directivas derivadas de una arquitectura de alto nivel, acompañada del soporte para la definición del proceso, la certificación de arquitecturas, y la capacitación relacionada con estas metodologías. Trabajar a partir de arquitecturas de software garantiza, al mediano plazo, la reducción de costos gracias al mejoramiento del tiempo de los ciclos productivos, la reducción en el nivel de defectos del software, la posibilidad de reutilización de las ideas subyacentes en la arquitectura y el diseño, y el mejoramiento en la comunicación a nivel técnico.

Si bien existe una abundante bibliografía de alta calidad dedicada a temas de seguridad y de arquitectura de software, ***la recopilación realizada en esta Tesis intenta abordar aspectos que intersectan ambos campos***.

Es frecuente que las organizaciones que desean incluir su política de seguridad en sus directivas de arquitectura enfrentan serias dificultades en diagramar un camino a seguir al momento de implementar las decisiones tomadas. A menos que se tome conciencia que el problema no reside en el talento y la competencia del equipo de desarrollo sino, más bien, en la falta de un *background* acerca de la seguridad, se deberá enfrentar una renovada resistencia, proyecto tras proyecto, al caer sistemáticamente en los mismos temas de discusión relacionados con la seguridad al momento de realizar la revisión a nivel de la arquitectura. Es por ello que urge contar con la capacidad de presentar estos temas de discusión dentro de un *framework* que ayude a conducir el proyecto.

Como arquitectos de sistemas, resulta deseable pensar que todas las decisiones que se toman están guiadas por consideraciones tecnológicas y objetivos del negocio. También es deseable creer que todas y cada una de las decisiones que son tomadas por el equipo de proyecto resultan consistentes con las directivas a nivel arquitectónico y sin verse afectadas por quién sea el integrante del equipo que las tome. Sin embargo, la naturaleza humana y la experiencia personal también influyen en las decisiones. En un sistema en construcción que debe respetar restricciones de presupuesto y de tiempo, la fortaleza de los líderes de diseño y de implementación puede orientar la dirección y la prioridad de los objetivos funcionales y no-funcionales[35].

[35] Un gurú en la metodología orientada a objetos podrá dedicar una significativa cantidad de recursos al desarrollo del modelo de datos y diagramas de clases; un programador con una importante experiencia en la construcción de código concurrente podrá introducir *multi-threading* en cualquier componente, creando "productores" y "consumidores" con los que hacer malabarismos con variables de condición en el diseño; un diseñador de base de datos de vasta experiencia en un producto podrá preconcebir nociones de cómo deberían ser las decisiones de diseño al momento de utilizar otra base de datos; un experto en CORBA podrá diseñar definiciones de interfaces o servicios con todo tipo de componentes que uno pudiera imarginar; un diseñador Web podrá lograr resultados altamente atractivos a nivel de la interface se usuario. Ninguna de estas acciones son inherentemente malas en sí mismas, sino más bien muy valiosas y

¿Qué ocurre si nadie en un equipo de trabajo posee experiencia en temas de seguridad? Frente a un conflicto entre un área en la que se trabaja con un alto grado de incertidumbre debido a esta carencia, y otra área en la que el equipo puede ejecutar una significativa cantidad de trabajo productivo, seguramente se ha de avanzar seleccionando aquellas tareas en las que se puede realizar el mayor progreso. El problema también se presenta con otras facetas de la arquitectura de sistemas, las que podrían ser dejadas de lado debido a la falta de experiencia o al hecho de no tener asignada alguna prioridad. El equipo puede declarar que no están lo suficientemente disponibles, que no se pueden testear de manera exhaustiva, o que no se pueden modelar debido a que se carece del tiempo o del dinero necesarios para llevarlo a cabo. Esta situación ocurre con más frecuencia de la que se sospecha y si nadie dentro del equipo posee alguna experiencia en la construcción de sistemas robustos, la naturaleza humana puede dirigir las decisiones descartando las tareas antes mencionadas.

La arquitectura de seguridad a menudo padece de este síndrome. Afortunadamente, empiezan a vislumbrarse soluciones para salvar esta brecha de conocimiento: buscar en el mercado de productos y contratar expertos para el aseguramiento del sistema, existente o en construcción. Pero este camino, recorrido por muchas organizaciones para atender a sus demandas de corto plazo, continúa exigiendo de una ***respuesta acorde al nivel de madurez alcanzado por la Ingeniería de Software, volviéndose parte integral de la gestión del proceso de desarrollo de software basado tanto en la calidad del proceso como en la calidad el producto***.

El pasaje que se produce desde la definición de requerimientos hasta la entrega del producto se puede considerar como una instancia de la aplicación de algún proceso de software. La organización que posea el proyecto y el sistema derivado del mismo también puede estar interesada en evaluar el éxito del proceso de software en sí mismo; un proceso exitoso ayuda a que el proyecto satisfaga los objetivos del clientes, conforme el presupuesto y el tiempo.

El proceso de software establece un conjunto de prácticas que proporcionan métodos medibles y repetibles para la incorporar la calidad en un producto de software. Como ya se indicara anteriormente, a medida que las organizaciones se esfuerzan en afrontar las complejidades del desarrollo de sistemas basados en software, la adquisición de recursos, la provisión de servicios, el desarrollo de productos, la operación de las infraestructuras, y la gestión de la evolución, la adopción de procesos de software comienzan a percibirse como un aspecto clave para poner orden en el caos. A su vez, los cuerpos de estándares han crecido en la construcción de frameworks en torno a la definición del proceso de desarrollo de software propiamente dicho.

Este proceso de desarrollo fluye en forma iterativa o incremental a través de una secuencia de

de evidente utilidad. Pero si al final de cuentas, el proyecto no satisface las necesidades del cliente con el adecuado nivel de desempeño y confiabilidad, el proyecto habrá fallado.

fases: factibilidad, definición de requerimientos, arquitectura, análisis, diseño, desarrollo, testeo, implantación y mantenimiento; la experiencia ha creado una amplia variedad de herramientas, procesos, y metodologías que asisten cada una de estas fases. Y cuando este proceso se vuelve formal gestiona la complejidad inherente, intentando guiar el orden de las actividades, las tareas relacionadas con el desarrollo, la especificación de los artefactos, y el monitoreo y medición de las actividades mediante el empleo de métricas concretas [Jacobson 99].

Existen muy diferentes corrientes en lo que hace al proceso de software, cada una con su propios fundamentos respecto de la naturaleza esencial del desarrollo de sistemas de software, teniendo actualmente un fuerte impacto en las prácticas adoptadas en nuestro medio:

- **Meta-procesos de software**: Miden la calidad, la aptitud, la competencia y la conformidad (*quality, capability, adequacy, conformity*) de las instancias particulares de los procesos de software utilizados en una organización; dichos meta-procesos reconocen factores de éxito críticos en la definición del proceso de softeware y en la medición del éxito de un proyecto en satisfacer estos factores, y un factor crítico para el éxito es la validación del documento de arquitectura del sistema. Son ejemplos el *Capability Maturity Model* –CMM– del *Software Engineering Institute* y estándares tales como *Software Process Improvement and Capability dEtermination* –SPICE– que proporcionan modelos para la definición de *frameworks* que guían a las organizaciones a través del proceso de implantación, valoración, medición, mejora y certificación.
- **Procesos de software**: Definen metodologías para la construcción de sistemas de software complejos. Es ejemplo el *Rational's Unified Process*, construido a partir del principio de diseño incremental manejado por casos de uso y centrado en la arquitectura, que evoluciona a lo largo de cuatro fases.

La mayoría de las definiciones de los procesos de software ubican a la seguridad en la misma clase que otros requerimientos no-funcionales del sistema, tales como realización, disponibilidad, portabilidad, desempeño, y testeabilidad[36]. Sin embargo, ***la seguridad no cabe dentro de un sistema de la misma manera que estos otros requerimientos no-funcionales, y no puede ser tratada de una manera uniforme***. Esta situación es una causa fundamental de muchas de las dificultades asociadas al intentar introducir la seguridad dentro de una arquitecta de sistema.

La seguridad difiere de las demás propiedades de un sistema en los siguientes aspectos:

- *El cliente de la seguridad como una característica de un sistema es la seguridad de la organización en su conjunto, y no el proceso de negocio del sistema*. Si bien los

[36] "*realibility, availability, portability, performance, testeability*"

costos de un sistema no-seguro son pagados por el cliente real, éste no es la fuente adecuada para establecer las directivas referidas a los requerimientos del sistema.

- *Las actividades de hacking no poseen casos de uso*. La metodología de casos de uso resulta excelente al momento de describir qué es lo que debería hacer un sistema cuando los usuarios actúan sobre él. Hacer robusto un caso de uso también puede tener sentido, garantizando que cualquier usuario que desee modificar el sistema sea autenticado y autorizado por hacerlo. Sin embargo, todas las interacciones permitidas por el sistema podrían ser explotadas por un agente malicioso no son, ni pueden serlo, parte de algún escenario de casos de uso y no pueden ser capturadas al momento de la definición de los requerimientos. No existen *casos de abuso*; existen demasiadas variables. La correcta inclusión de la seguridad dentro de la arquitectura ha de requerir de las arquitecturas un mayor conocimiento en el dominio de la seguridad junto con la correspondiente experticia en responder a las intrusiones e incidentes. Ningún perfil operativo incluye el uso malicioso.
- *La aceptación de un sistema por parte de un cliente nunca puede ser considerada como la ausencia de bugs*. Edward Dijkstra ya estableció que "El testeo sólo puede probar la presencia de *bugs*, no su ausencia". No se puede testear lo que se desconoce.

La literatura relacionada con el proceso de software se mantiene en silencio sobre cómo gestionar la seguridad bajo circunstancias únicas. La literatura concerniente con la seguridad es muy rica, pero ofrece recomendaciones de arquitectura de software de tipo prácticas que guían las decisiones en la fase de definición de los componentes del sistema[37].

El resultado de una motivación que inicialmente estuvo orientada a comprender y analizar el *framework* "*Trustworthy Refinement Through Intrusion-Aware Design (TRIAD)*", condujo a rastrear sus antecedentes y, a partir de allí comenzar a vislumbrar la potencialidad de los trabajos realizados en los últimos años, y que continúan activamente, tanto en el campo de la Ingeniería del Software como de la Ingeniería de Requerimientos.

Fue esencial comprender que no existen respuestas fáciles que permitan conducir con éxito la producción de sistemas de información defendibles. Y que sólo es posible hacerlo abordando el estudio de herramientas que plantean el problema de la gestión de la seguridad desde un punto de vista estructural de los sistemas de información distribuidos. Esto constituye el paso inicial para superar estas realidades, ya que busca:

1. Hacer más explícitos los requerimientos de la supervivencia y sus propiedades in-

[37] La mayoría se focaliza en aspectos que atienden problemas de corto plazo presentes al momento de desplegar el sistema, tales como una correcta configuración y despliegue de componentes de seguridad, el empleo de herramientas de auditoría, los sistemas de detección de intrusión, *firewalls*.

herentes necesarias, tales como seguridad, confiabilidad y rendimiento, y la caracterización de las interacciones entre los diferentes sub-requerimientos.

2. Identificar la funcionalidad cuya ausencia impide la adecuada satisfacción de aquellos requerimientos y recomendar el desarrollo de componentes específicos de la infraestructura faltantes o que no se encuentran disponibles comercialmente en la actualidad.
3. Explorar técnicas para el diseño y desarrollo de sistemas y redes con alto grado de supervivencia, a pesar de la existencia de sub-sistemas y personas no dignos de confianza –en donde esta falta de confianza puede incluir falta de confiabilidad, integridad y corrección en el comportamiento de los sistemas y las personas.
4. Recomendar arquitecturas estructurales específicas que puedan conducir a sistemas y redes con capacidad de supervivencia en condiciones de prevenir o de tolerar una amplia variedad de amenazas.
5. Explorar principios operacionales que pueden mejorar la capacidad de supervivencia.

Por otra parte, se analizó el documento "*Common Concepts Underlying Safety, Security, and Survivability Engineering*", el cual presenta un consistente conjunto de modelos de información que identifican y definen conceptos básicos en el campo de la ingeniería de seguridad y supervivencia, y donde se intenta demostrar que los requerimientos de calidad están relacionados con los factores, criterios y métricas de calidad, poniendo énfasis en las similitudes entre los conceptos básicos que sirven de fundamento a los mencionados campos. Los modelos de información propuestos proveen una terminología estandar y un conjunto de conceptos que explican las similitudes entre los métodos basados en la evaluación de activos y determinación del riesgo en la identificación y análisis de requerimientos como así también en las similitudes de los componentes arquitectónicos (componentes) más utilizados para satisfacer a estos requerimientos.

9.1.1 Los antecedentes del Modelo TRIAD

Se tomó como base el Plan de Trabajo del CERT en el campo de los Sistemas con Capacidad de Supervivencia, cuyo objetivo global es el de mejorar las prácticas de la ingeniería en lo que respecta la supervivencia a partir de sólidos fundamentos ingenieriles, y como objetivos particulare el desarrollo de fundamentos teóricos, representaciones de lenguaje y métodos unificados, rigurosos pero prácticos, para representar y razonar estos sistemas, sus componentes (a menudo COTS) y sus ambientes de amenazas.

Empleando los resultados de estos trabajos, comienza a ser posible aplicar la premisa de que en cada actividad del ciclo de vida de un sistema de información se deberán considerar los objetivos de defendibilidad, incorporando los métodos que aseguren este requerimiento de calidad, evitando relegar sus aspectos específicos a un hilo separado de la actividad del proyecto que

conduce a que ésta termine siendo tratada como una propiedad extra. Los aspectos relacionados con la defendibilidad necesitan ser establecidos en forma temprana, como por ejemplo, durante la conceptualización de las operaciones, y revisados en las principales etapas del desarrollo, tal como al momento de definir los requerimientos, la arquitectura, etc. Esto pareciera sugerir el empleo de un ciclo de vida tipo cascada, pero de hecho opera muy adecuadamente con los modernos modelos de ciclo de vida tal como el modelo en espiral.

Asímismo, es posible aplicar la experiencia adquirida en la aplicación de métodos de desarrollo existentes, tal como el Método *Survivable Network Analysis* SNA, que ha venido siendo utilizado durante varios años con los clientes del CERT. Si bien recién se comienzan a tener algunos datos de costo/beneficio, en la mayoría de los casos resulta claro que la recomendaciones surgidas de su aplicación han mejorado la defendibilidad de los sistemas de sus clientes.

SNA es un proceso estructurado que aplica el modelo en espiral, orientado al mejoramiento de las características de defendibilidad de sistemas (tanto nuevos como existentes) que se desenvuelve a lo largo de una serie de sesiones de trabajo conjunto entre especialistas y personal del usuario, y cuyas conclusiones se resumen en un reporte destinado a gestionar acciones.

Este proceso comienza con los informes de los usuarios del sistema, los colaboradores y desarrolladores, los cuales generalmente están centrados en el nivel de la arquitectura. El proceso de descubrimiento continúa con las visiones que poseen el desarrollador, el usuario y los responsables de los servicios esenciales del sistema; estos servicios se formulan como escenarios de uso paso-a-paso y se los rastrea a lo largo de la arquitectura para poner de manifiesto los componentes esenciales. Luego se identifican las intrusiones representativas sobre la base del análisis del ambiente de amenza, y en forma análoga, se los expresa como escenarios de uso para rastreabilidad a lo largo de la arquitectura, para así descubrir los componetes que pueden ser comprometidos. Con esta información, resulta posible identificar aquellos componentes que representan puntos débiles, es decir, que son tanto esenciales como pasibles de verse comprometidos; continuándose con el análisis de supervivencia para introducir mejoras las estrategias de resistencia, reconocimiento y recuperación dentro de la arquitectura del sistema. A menudo resulta que las recomendaciones se propagan hacia áreas tales como requerimientos, políticas y operaciones.

La aplicación del SNA en clientes ha permitido la formulación de tres observaciones claves:

- Se requieren de métodos de evaluación sistemáticos para establecer la supervivencia de un componente COTS, debido a que muchas organizaciones se encuentran desarrollando sistemas de misión crítica en los que emplean componentes COTSs. Los COTSs suelen presentar menores costos que las soluciones hechas a medida, pero las organizaciones que los adquieren carecen de acceso a los artefactos del proceso de ingeniería de software utilizados para crear los componentes. El análisis de los artefactos de ingenie-

ría es el medio tradicional para verificar la supervivencia de los sistemas hechos a medida. Una forma de compensar parcialmente esta carencia es la utilización de herramientas de evaluación de riesgo para la construcción, mantenimiento y evolución de los sistemas con capacidad de supervivencia. El equipo de trabajo se encuentra desarrollando una solución orientada a la gestión del riesgo, denominada *Vendor Risk Assesment and Threat Evaluation – V-RATE*, destinada a la evaluación de los sistemas basados en COTSs. Esta herramienta ayudará a las organizaciones compradoras a comprender el balance asociado con el uso de productos COTSs, y la posibilidad de alcanzar los niveles de seguridad requeridos mediante la evaluación de diferentes diseños de sistemas basados en productos COTSs.

- Las complejidades de un sistema distribuido de gran escala se pueden reducir y administrar mediante una disciplina de ingeniería unificada destinada al análisis y el diseño, que incluya la capacidad de supervivencia dentro de un *framework* integral. Las complejidades en el análisis y diseño de este tipo de sistemas a menudo exceden las capacidades de la ingeniería en lo que se refiere a su control intelectual. El equipo de trabajo se encuentra definiendo los fundamentos ingenieriles para la tecnología *Flow-Service-Quality – FSQ*, que se basa en estructuras de flujo de tareas de usuario y en su rastreabilidad a través de la arquitectura, una aproximación computacional para calificar atributos (incluida la supervivencia), y un *framework* arquitectónico para la administración dinámica de los flujos y de sus atributos de calidad. Este proceso se puede aplicar tanto a la especificación, diseño y operación de sistemas nuevos, como al análisis de sistemas existentes en lo referente a dependencias y riesgos de supervivencia que puedan impactar en el desempeño de la misión. El mismo también ayuda a la integración de sistemas *stovepipe* pre-existentes para soporte de nuevos objetivos de misión.
- La documentación estructurada y el uso sistemático de patrones de ataque y estrategias de supervivencia pueden ayudar al diseño y el análisis de arquitecturas resistentes a la intrusión. La importante inversión en tecnología de seguridad de la información por parte de una organización a menudo se ve escasamente reflejada, y por ende resulta cuestionada, como un valor para la misión operativa. Una razón importante es que gran parte de los esfuerzos en el diseño y el análisis se centran en la decisión de cuáles tecnologías de seguridad más populares se han integrar, en lugar de realizar una evaluación racional acerca de la manera de enfrentar los ataques que son de esperarse se produzcan para comprometer la misión. El trabajo en equipo incluye la incorporación de técnicas de análisis de intrusión y de riesgo a las prácticas de desarrollo existentes. Este trabajo requiere de la consideración de un amplio contexto operativo en el cual reside la tecnología, el cual se ha denominado

la organización. Las arquitecturas de la organización necesitan ser desarrolladas y analizadas de la misma manera que los sistemas en los que están basadas.

9.1.2 El Método V-RATE

El uso de productos COTS en el diseño y desarrollo de sistemas de software de gran escala (tanto de misión crítica como en subsistemas de respaldo de los mismos) presenta una extendida popularidad en la última década, y puede observarse que los diseñadores e implementadores se inclinan por el empleo de este tipo de productos debido a una diversidad de razones, entre las que se incluyen reducción de costos, economía en los tiempos de planificación, implementación e implantación de la solución, o en la falta de la suficiente experticia para resolver un problema particular.

Incluso los organismos gubernamentales (relacionados con sistemas de defensa y seguridad) así como empresas de gestión de sistemas críticos a escala nacional o regional se apoyan cada vez en su empleo, lo que debería plantear la consideración de nuevos temas de discusión. Tal el caso de cómo los ingenieros de sistemas (en todos los planos de la actividad) deberán construir los sistemas de aquí en más a partir de componentes en cuya especificación de atributos de calidad claves (por ejemplo, funcionalidad, confiabilidad, etc.) no han tenido participación. Esta situación dista significativamente de aquélla en que los componentes eran casi completamente a-medida, y en los que los desarrolladores poseen una visibilidad total sobre cada porción de software que es utilizado, y en los que el estilo de desarrollo, los resultados de los testeos, los reportes de bugs y el acceso al código fuente están todos al alcance.

Con el advenimiento del empleo de componentes COTS en los sistemas de misión crítica, muchas de las ventajas están desapareciendo. De hecho, existe un reconocimiento que tiende a generalizarse respecto a que el desarrollo de un sistema basado en COTS se presenta altos niveles de riesgo.

Esta situación plantea un interrogante de interés: ¿de qué manera un desarrollador de un sistema de misión crítica con tolerancia a falla puede saber si un producto COTS particular es adecuado para ser empleado en ese tipo de ambiente? El problema va más allá de la "simple" pregunta de si un producto satisface o no los requerimientos técnicos o económicos, ya que en un sistema de misión crítica se necesitan tener un claro conocimiento de aspectos tales como si un producto es estable, confiable y robusto.

El Método V-RATE fue desarrollado para ayudar a los responsables del diseño e implementación de sistemas puedan examinar los riesgos asociados con el empleo de componentes COTS en sistemas de misión crítica. V-RATE es un método basado en una taxonomía destinada a identificar los riesgos relativos al fabricante; examina no sólo los riesgos asociados con el fabricante

de un producto particular sino tambien la capacidad de la organización involucrada de afrontar estos riesgos. Además de atender a la compentecia técnica del componente COTS, la taxonomía considera problemas tales como la visibilidad sobre el proceso de desarrollo del fabricante, la confianza que se puede depositar en el mismo, y el nivel de conformidad con certificaciones de terceras partes.

La taxonomía V-RATE aún es relativamente nueva y, de acuerdo a los autores del método, todavía se ha podido recolectar poca información sobre su aplicación.

En la actualidad existen varios métodos destinados a gestionar la selección y uso de componentes COTS en proyectos basados en sistema de información. Son ejemplos:

- *Evolutionary Process for Integrating COTS-Bases Systems* [Albert 02] -EPIC-, basado en un proceso de "mejores práctica" a seguir para la integración de COTS en el desarrollo de un proyecto.
- *Procurement-Oriented Requeriments Engineering* [Maiden 98] -PORE-, *Off-The-Self-Option* [Kontio 96] -OTSO-, y *Component Evaluation Process* [Polen 99] -CEP-, que ponen el acento en la selección de productos COTS apropiados.
- *COTS Usage Risk Evaluation* [SEI 02] -CURE-, basado en el uso de un cuestionario temprano durante el proceso de desarrollo destinada a identificar, y luego mitigar, potenciales riesgos.
- *COCOTS* [Boehm 00], modelo en el que se examinan algunos de los riesgos asociados con el desarrollo basado en COTS desde la perspectivas de los costos.

V-RATE difiere de la mayoría de los métodos anteriores en dos formas significativas:

1. Intenta tratar de manera específica aspecto que son de importancia para los desarrolladores de sistemas de misión crítica con capacidad de supervivencia.
2. V-RATE no intenta atender a múltiples diferentes aspectos de la selección de productos COTS, su integración e testeo, como muchos de los otros métodos lo hacen.

Más bien, V-RATE se centra específicamente en la evaluación de los riesgos asociados con el empleo de un producto de un fabricante particular. Debido al estrecho y bien focalizado alcance, basado en una taxonomía detallada, V-RATE aparece como un candidato fundamental para ser empleado en la identificación y mitigación de los riesgos asociados con el empleo de COTS en proyectos de desarrollos de misión crítica.

La taxonomía V-RATE comprende dos secciones principales. La primera, denominada "Elementos de Riesgo Inherentes al Fabricante", lista los aspectos que son específicos del fabricante que deben ser examinados, los cuales se dividen a su vez en siete categorías principales.

- *Visibilidad de los Atributos del Producto*. ¿Están disponibles para su examen los artefactos del proceso de ingeniería de software empleados por el fabricante?
- *Aptitud Técnica*. ¿El fabricante respecta procesos formales de ingeniería de software de aseguramiento de la calidad, y sus productos conforman los estándares de aplicación?
- *Historia de Desempeño*. ¿El fabricante posee productos que se han demostrado un adecuado desempeño en aplicaciones de misión crítica?
- *Conformidad*. ¿El fabricante desea trabajar con los clientes en cuestiones relacionadas con la seguridad y la supervivencia?
- *Fidelidad*. ¿El fabricante posee una buena reputación en lo que hace a la provisión de productos de alta calidad y soporte equivalente?
- *Compentencia en la Gestión del Negocio*. ¿El fabricante es una empresa bien gestionada y económicamente viable?
- *Evolución Controlada*. ¿Los productos de fabricante evolucionan de una manera bien definida?

La segunda, denominada "Elementos de Riesgo del Fabricante Asociados con el Experticia del Especialista en la Gestión del Riesgo del Fabricante", examina los riesgos que tiene que ver con las capacidades de gestión del riesgo propias de una organización, la que consta de seis categorías.

- *Factores Técnicos de Mitigación del Riesgo*. ¿La organización a la que pertenece el especialista posee habilidades en la evaluación de la calidad de productos y de la compentencia de fabricantes, o posee experiencia en la integración de productos COTS en ambientes de misión crítica?
- *Factores No-Técnicos de Mitigación del Riesgo*. ¿Existen aspectos económicos, legales, entre otroas, tales como seguros, contratos de niveles de servicio, o garantías que protejan al especialista?
- *Independencia / Interdependencia*. ¿En qué grado una parte del producto del fabricante depende de algún otro actor, y de qué manera esto afecta a la supervivencia?
- *Grado de Exposición de la Organización*. ¿En qué pérdidas incurriría el especialista debido a una falla en el producto del fabricante?
- *Alineamiento con la Misión / Compatibilidad del Fabricante*. ¿El producto del fabricante exhibe los atributos de calidad necesarios para que el sistema en desarrollo lleve adelante su misión?
- *Experticia en la Negociación / Poder de Negociación*. ¿Qué influencia posee el especialista sobre el fabricante, sea ésta económica o de otro tipo?

Mediante el examen de cada elemento de la taxonomía V-RATE, y el desarrollo de las estrate-

gias de mitigación para cada riesgo identificado, un especialista en aseguramiento crea lo que se denomina un "Perfil de Riesgo del Fabricante", y que luego es utilizado para identificar el nivel de riesgo inherente al uso del producto del fabricante como parte de un sistema, como así también con cuáles áreas, si las hubiera, han de requerir atención como parte del desarrollo.

De todas maneras, quedan abiertos a los experiencias de utilización de esta taxonomía, una lectura crítica sobre el nivel de detalle con que están descriptos los elementos incluidos en la misma, dado que algunas descripciones presentan significados casi similares. El poder contar con una versión detallada de la taxonomía que incluya ejemplos de cada elemento podría eliminar ambigüedades y hacer más sencilla su empleo en las primeras experiencias. Incrementar la especificidad mejoraría la confianza en la taxonomía de V-RATE, con una potencial reducción del tiempo y esfuerzos requeridos para su uso.

Otro aspecto que aparece como una tarea nada fácil es la de poder incluir información útil en el Perfil de Riesgo del Fabricante, y una de las principales razones de la poca disposición de los fabricante en compartir conocimientos, especialmente en aquellas áreas que tienen que ver con los procesos de desarrollo, testeo por terceros independientes, y futura evolución del producto. Esta reticencia puede deberse a múltiples razones, tales como si la organización es un cliente importante para el fabricante, el riesgo que representa para el fabricante hacer pública información vía el perfil en construcción, o si el fabricante posee respuestas satisfactorias para algunos de los elementos de la taxonomía. Más allá de la causa, si el fabricante no es capaz de entregar información pertinente al especialista, el perfil resulta carecerá de especificidad y resultará en una herramienta de menor utilidad.

Más allá de las dificultades que se puedan encontrar en la obtención de la cooperación del fabricante durante la generación del perfil, resultan evidentes las ventajas de seguir cada uno de los pasos indicados en la taxonomía V-RATE, en particular para el caso de los elementos de la segunda sección de la taxonomía. En la gran mayoría de los casos, los componentes COTS son seleccionados en base a funcionalidades, sin tener un peso significativo las condiciones posventa. Sin embargo, la experiencia que puede adquirirse al ejecutar los elementos de la segunda sección es digna de una especial atención; un ejemplo claro resulta especialmente de la sección Factores Técnicos de Mitigación, cuyo propósito va más allá del Método V-RATE, tal como la identificación de potenciales áreas de futura capacitación.

Además de los usos indicados anteriormente, V-RATE también posee otros potencialmente significativos beneficios. Los autores aspiran a la acumulación de una importante colección de perfiles, a fin del seguimiento real del desempeño del fabricante sobre una diversis de proyectos de desarrollo del mundo real. Estos perfiles servirían como una referencia contra la cual un especialista podrá comparar perfiles más recientes. Esta capacidad de comparar Perfiles de Riesgo

de Fabricante contra una base de datos de perfiles existentes pondría a disposición de los especialistas una herramiento con la cual medir el riesgo relativo de utilizar un producto COTS particular, comparación que resultaría de especial utilidad en el caso que el fabricante ya posea perfiles en la base de datos.

De resultar posible establecer un método para compartir información potencialmente confidencial, el contenido de la base de datos de perfiles podría facilitar el desarrollo de estrategias comunes de mitigación para hacer frente a los riesgos delineados en la taxonomía V-RATE. A lo que se suma que tal base de datos podría, eventualmente, motivar la creación para parte de un tercero independiente, de porciones "estandar" de perfiles para diferentes fabricantes que tiene que ver con la sección de Elementos de Riesgo Inherentes al Fabricante.

9.1.3 El *Framework* FSQ

El desarrollo de sistemas basados en red de gran escala que satisfagan altos niveles de requerimientos en cuanto a aseguramiento deben resolver temas de ingeniería desafiantes. La complejidad de los sistemas que día a día emergen para satisfacer las demandas de nuesta sociedad es tal que excede muchas veces las capacidades de control intelectual, causando retrasos y frustraciones durante el proceso de desarrollo, y errores y compromisos de seguridad durante la operación. El *framework* FSQ provee los fundamentos y las metodologías de orden práctico para atender a estas demandas que hacen factible mantener el control intelectual durante el análisis de sistemas, su desarrollo y mantenimiento.

Los aspecto más complejos y conflictivos presentes en estos sistemas son:

- Topologías heterogéneas con límites, componentes y usuarios altamente cambiantes
- Inteconexión dinámica que limita la visibilidad y el control
- Flujos de tareas a nivel de usuario que atraviesan componentes cuya seguridad y confiabilidad a menudo son desconocidas
- Funcionamiento y calidad de componentes COTS inciertos
- Operaciones fuertemente asincrónicas que son un reto para la comprensión humana
- Requerimientos por parte de las organizaciones de un desarrollo rápido y con un alto nivel de aseguramiento

Dadas estas realidades, ¿qué fundamentos de ingeniería se pueden definir para mantener el control intelectual? No es posible focalizarse en una ingeniería de componentes, debido a que los componentes por sí mismos son inadecuados de definir el nivel de comportamiento de un sistema requerido por una organización. Más bien, los fundamentos deben ser capaces de embeber el refinamiento de las misiones y tareas a nivel organización dentro de estructuras de sistemas en red, que sean las que describan las especificaciones de componentes, los atributos de calidad y

los procedimientos operacionales. Este proceso no debe presentar fisuras y ser independiente de la escala, y atender directamente a las demandas de los sistemas basados en red enunciados al inicio que muchas veces son la causa que frustran los objetivos de alto nivel de aseguramiento.

En este sentido, el *framework* FSQ constituye un campo emergente orientado a dar soporte al proceso de análisis, especificación, diseño, validación, implementación y operación basado en estas necesidades.

La ingeniería FSQ comprende tres tecnología integradas: Estructuras de Flujo, Atributos de Calidad Computables y Arquitecturas de Gestión de Flujo.

Bajo este *framework*, los sistemas basados en red que deben su existencia a la satisfacción de requerimientos de la misión de una organización, están instanciados en conjuntos de flujos de tarea de usuario. A su vez, estos flujos se pueden refinar en composiciones de servicios de red provistos por los componentes de hardware, software y humanos, todos sujetos a requerimientos de atributos de calidad.

Estas estructuras son definiciones de las etapas y de las decisiones aplicadas a las tareas de usuario, y su refinamiento en usos de servicios del sistema. Las estructuras de flujo son artefactos estables que poseen semánticas definidas a fin de mantener el control intelectual en medio de la incertidumbre del desarrollo y de la evolución de un sistema en red. Se las puede expresar como estructuras de control simples, y pueden ser refinadas, abstraídas y verificadas con precisión.

Los flujos invocan servicios de sistemas, los cuales pueden, a su vez, ser refinados en forma de flujos, etc., conformando procesos recursivos que emplean métodos idénticos en todos los niveles de diseño.

De esta manera, la visión que se posee de los sistemas basados en red es la de topologías de componentes comunicándose de manera asincrónica, que presentan el comportamiento esperado, si bien ocasionalmente pueden no hacerlo, que pueden estar compuestos de diferentes maneras, a fin de satisfacer los flujos de tarea de usuario. Tal impredectiblidad, sea debida a errores, fallas o intrusiones, es una característica perversa de los sistemas basados en red, y un problema de gestión de riesgo a nivel de la organización con consecuencias potencialmente serias.

Se han definido semánticas matemáticas de las Estructuras de Flujo a fin de permitir, como una práctica corriente de la ingeniería, el desarrollo y la verificación de flujos de cara a las incertidumbres. Para atender a un comportamiento impredictible de los servicios, dichas semánticas requieren la especificación de sólo el procesamiento que el flujo mismo lleva adelante, y no el procesamiento de los servicios que invoca. Resulta innecesaria la especificación de qué es lo que hacen los servicios (lo que puede resultar desconocido), sólo qué es lo que hace un flujo con sus respuestas, esperadas o inesperadas. Es semántica basada en respuestas hace que los flujos

puedan exhibir un comportamiento determinístico accesible a la comprensión humana, a pesar del asincronismo subyacente de los usos de servicios compartidos.

Las semánticas de las Estructuras de Flujo están basadas en un modelo teórico que trata a los flujos y a sus estructuras de control como reglas que se traducen en funciones o en relaciones matemáticas. A continuación se resumen los requerimientos de dichas semánticas conforme a las demandas de los sistemas en red:

- Visión basada en tareas de la organización de los sistemas en red de la QoS → Las semánticas tratan a los flujos de tareas como los principales artefactos para la especificación, diseño y uso de un sistema
- Refinamiento sin fisuras de parte de la misión de la organización hasta alcanzar la implementación de los sistemas → Las semánticas soportan el refinamiento de la misión en flujos de tareas, y luego en usos del servicio del sistema
- Conocimiento incompleto del comportamiento de los servicios en red → Las semánticas de flujo están basadas sólo en las respuestas del servicio, y no en el comportamiento global del mismo
- Incertidumbre en la función y los atributos de calidad de los servicios en red → Contar con semánticas que requieran el diseño de incertidumbres debidas a fallas y compromisos
- Complejidad de los comportamientos perversos asincrónicos en las operaciones de red → Las semánticas pueden ser determinísticas para la comprensión y análisis humano
- Complejidad en la especificación y diseño de la topología de red → Semánticas que definen en forma suficiente la topología de red como la unión de todos los flujos
- Complejidad en la especificación y diseño del servicio de sistema → Semánticas que definen los requerimientos de servicio como la unión de todos los usos establecidos por los flujos

Las Estructuras de Flujo exhiben propiedades deseables para el desarrollo de los sistemas bajo consideración. Se cuenta además con teoremas de verificación de flujos que definen las condiciones para la verificación de éstos con respecto a sus efectos esperados. Se pueden emplear flujos a nivel de organización para la definición y validación de los procesos de negocio globales; estos flujos principales ponen de relieve, a su vez, los requerimientos para los flujos secundarios, y así sucesivamente, en un proceso de cierre transitivo que ayuda a garantizar la integridad de las especificaciones del sistema.

Resulta altamente recomendable encarar el diseño inicial de la topología de la red en base a la combinación de los flujos resultantes, y la especificación de cada servicio como la combinación de todos sus usos existentes en los flujos.

Por otra parte, el *framework* FSQ trata a los atributos de calidad como aquellas propiedades a ser definidas, computadas y realizadas como características dinámicas de los sistemas, cuyos valores cambian constantemente durante la operación de los mismos. Es decir, los atributos de calidad son tratados como funciones a ser computadas, y no sólo como descripciones estáticas y a-priori de las propiedades a ser alcanzadas.

En tanto tales funciones descansan en aspectos como qué puede ser computado y, en consecuencia, difieren de los métodos tradicionales, permiten una nueva manera de encarar el análisis, diseño y evaluación de atributos. Los requerimientos a nivel de atributos se pueden asociar con los usos del servicio de sistema embebidos dentro de los flujos. El requerimiento de que los atributos sean mensurables en métricas definidas para el cómputo también permite un análisis humano que sería imposible de otra manera.

Estos Atributos de Calidad Computables son otro artefacto de primer orden en el *framework* FSQ, los que encarnan un modelo de atributo funcional que asocia información relativa a la utilización del servicio con valores del atributo, un modelo de transición de estado para la evaluación de atributos, y métodos basados en el Teorema de Bayes para la evaluación de atributos dinámicos.

Finalmente, las Estructuras de Flujo y los Atributos de Calidad Computables posibilitan la consideración de arquitecturas de sistema que implementan dinámicamente los flujos y la gestión de atributos durante la ejecución. Se espera poder contar en el futuro con plantillas de topologías de sistema y capacidades funcionales para la gestión de instanciaciones de flujos y la compatibilización de sus requerimientos en cuanto a atributos de calidad con el comportamiento y las capacidades del servicio bajo una operación en tiempo real.

El *framework* FSQ recibe el aporte proveniente de diferentes áreas, tales como:

- *Workflows*: en la actualidad, éstos son un método principal para la especificación de los requerimientos de negocio [Leymann 00b. Schmidt 99], que pueden ser modelados gráficamente como una diagrama de trazas que describe el orden en que se pueden ejecutar los servicios. Por ende, un flujo FSQ es análogo a un *workflow* especificado de tal manera que presenta los servicios y atributos embebidos, permitiendo contar con un recurso más de ingeniería.
- Casos de Uso: son una notación gráfica para que permite capturar las interacciones usuario-sistema [Jacobson 92], de amplia aplicación como aprte del *Unified Modeling Lenguage* – UML. Si bien resultan de gran utilidad durante el trabajo con el usuario, son bien conocidas sus deficiencias que impiden su uso como un lenguaje formal de especificación de requerimientos [Glinz 00].
- Sistemas basados en componentes: una corriente importante en el desarrollo de sistemas

es el *desarrollo de software basado en componentes* [Brown 98]. El uso efectivo de los conceptos de arquitectura de software provee el basamento para la construcción de sistemas extensos a constituidos por pequeños componentes del sistema pre-desarrollados; los componentes son seleccionados y conectado para proveer la funcionalidad requerida por las reglas de negocio del sistema. Un objetivo de este tipo de desarrollo es el de tener un contexto abierto para que los fabricantes desarrollen componentes que podrán ser integrado en modalidad *plug-and-play* dentro de arquitecturas de sistemas abiertas. En el caso del *framework* FSQ, los servicios resultan análogos al concepto de componentes; si bien algunos trabajos de investigación han encarado combinar servicios con atributos de calidad [Yacoub 99], no existe un *framework* sistemático para llevarlo a cabo. Los métodos que intentan describir de manera rigurosa el comportamiento a nivel de sistema y las propiedades de calidad por lo general no escalan adecuadamente en la dirección de sistemas con el tamaño y la complejidad de sistemas críticos para la infraestructura [Sullivan 99]. Generalmente, la descripción de un sistema se realiza con un alto nivel de abstracción y proporcionan un insuficiente rigor para el análisis cuantitativo en el nivel de detalle necesario para comprender si el sistema puede satisfacer las calidades de los servicios esenciales de bajo nivel.

- *Open Distributed Proccesing*: este término hace referencia al *framework* estandar de la ISO para el desarrollo y la operación de sistemas distribuidos [Joyner 01]. El Modelo de Referencia ODP está formado por un conjunto de *viewpoints* que definen *funciones* que implementan *transparencias*. ODP es un esfuerzo que está teniendo lugar orientado a proveer fundamentos unificadores para los muchos estandares relativos a sistemas distribuidos que existen actualmente. El objetivo de FSQ también es el de proporcionar un modelo más limpio y ordenado para el desarrollo de sistemas de gran escala que incluye la calidad como un concepto clave.
- Coordinación de Sistemas Basados en Agentes: los trabajos de investigación en el campo de los sistemas basados en agente atienden al problema de la coordinación e integración en el contexto de sistemas distribuidos de gran escala empleando un *framework* multi-agente [Sikora 98]. Esta formalismo representacional se utiliza como base para la integración de una colección de sistemas de información autónomos, cada uno de los cuales es visto como un agente. Se pueden utilizar diferentes tipos de mecanismos de corrdinación y esquemas de control para la integración de diferentes unidades de sistema de información. Los niveles de atributos de calidad del sistema pueden ser evaluados en base a la interacción de estos agentes.

Podemos concluir afirmando que el *framework* FSQ es fundamentalmente diferente a otros *fra-*

meworks en el sentido que se focaliza en la capacidad de un sistema de satisfacer las solicitudes de servicio particulares en base a niveles de calidad especificados. Descompone los sistema jerárquicamente en términos de las solicitudes de servicio de usuarios, en lugar de hacerlo en términos de funciones u objetos. El control de un sistema distribuido de gran escala queda reducido a un proceso sistemático de gestión de flujos de usos de servicio y de evaluaciones de calidad, gestión que puede ser centralizada o descentralizada, dentro de la estructura del sistema. El análisis de los atributos de calidad se reducen a un análisis sistemático de flujos de servicios de usuario, y la computación y composición de propiedades de supervivencia con respecto a modelos dinámicos de flujos bajo operación.

9.2 CONCLUSIONES ACERCA DEL TRIAD

En este trabajo se realizado la descripción de un modelo de diseño intrusión-aware, denominado TRIAD, para el refinamiento sistemático de arquitecturas de sistemas de información en dominios complejos, potencialmente sin límites definidos, para resistir, reconocer, ser capaces de recuperarse y adaptarse a patrones de ataques conocidos e hipotéticos.

TRIAD facilita la planificación de los cambios inevitables sobre el ambiente de amenazas y operacional y ayuda a la trazabilidad del efecto de los cambios frente a requerimiento de supervivencia y de arquitectura. La estructura en espiral itera a lo largo de tres sectores de actividad para el desarrollo de la estrategia arquitectónica, para la instanciación de la arquitectura utilizando componentes técnicos, y para analizar el impacto del ambiente de amenazas sobre las operaciones del sistema. A continuación se describen las actuales limitaciones del modelo TRIAD, de qué manera se puede utilizar TRIAD dentro de un contexto más abarcativo de los ciclos de vida de desarrollo de sistemas, y los trabajos futuros para un mayor refinamiento y aplicabilidad de TRIAD en ejemplos más complejos del mundo real.

9.2.1 Utilización del Modelo

TRIAD se ocupa de sólo una pequeña, pero importante, parte del ciclo de vida de desarrollo de sistemas con capacidad de supervivencia. En particular, el modelo no se ocupa específicamente de:

- La implementación, evolución o mantenimiento de la arquitectura de supervivencia derivada.
- Las funciones o propiedades requeridas o deseadas del sistema que no contribuyan con la misión.
- Las fallas relevantes de la supervivencia debidas a fallos o accidentes internos.
- Los riesgos de programa, tales como deficiencias del equipo de desarrollo, que no son debidos a una actividad maliciosa.

La incorporación de TRIAD en el proceso de desarrollo y mantenimiento de sistemas (SDM – System Development and Maintenance) requerirá de la resolución de muchos de estos problemas. Una aproximación detallada de cómo hacer esto depende en gran medida de los detalles del dominio de problema del sistema y del ambiente de desarrollo, lo que excede al alcance del trabajo bajo análisis. A pesar de ello, discutiremos algunos de los problemas relacionados con la ***provisión de una base para la formulación de un proceso SDM comprensivo que incorpora conceptos IAD***. Afortunadamente, los modelos en espiral iterativos son útiles tanto para la caracterización del mantenimiento del sistema (o su mejora) como para el desarrollo del sistema [Boehm 88].

Como se mencionara anteriormente, el desarrollo de sistemas de sistemas de información de alta confidencialidad con configuraciones complejas, en los que el impacto de las fallas debidas a intrusión es severo demanda de un proceso manejado por riesgos del tipo del modelo en espiral para resolver en forma gradual la incertidumbre de la manera más eficaz posible. El empleo de TRAID dentro del contexto de una espiral SDM comprensiva puede llevarse a cabo en dos formas (ver Figura 61):

- *Ver a TRIAD directamente como una mini-espiral. En este caso, la ejecución del proceso IAD conduce a un punto de arranque avanzado de la espiral más amplia SDM.*
- *Desplegar las actividades y las estructuras documentadas TRIAD dentro de los primeros ciclos de la espiral SDM. En este caso, se produce una integración más compresiva de los dos procesos.*

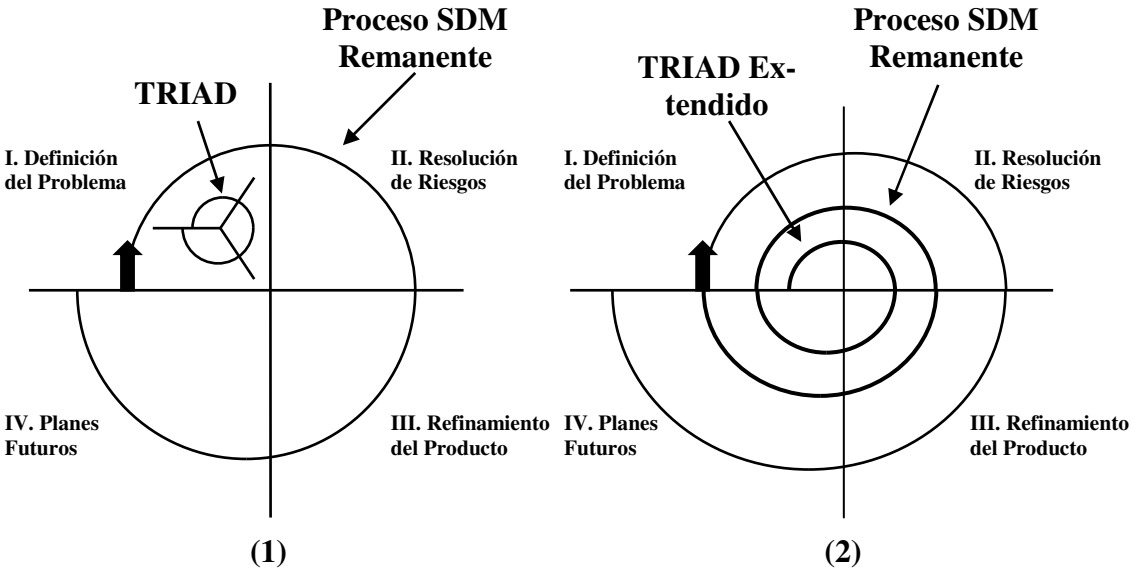


Figura 61. TRIAD dentro del proceso SDM como (1) una mini-espiral (2) integrado.

El primero de estos métodos es una alternativa viable debido a que TRIAD se centra en la mi-

sión. Pensamos que los requerimientos de supervivencia relacionados con la misión se deben emplear para determinar el formato general de la arquitectura y debe, en consecuencia, ser el foco de las iteraciones iniciales del proceso de diseño. Las funciones o propiedades requeridas o deseadas que no contribuyen a la misión deben ajustarse dentro de los parámetros definidos por la arquitectura de supervivencia y no deben disminuir significativamente la confianza que los dueños del sistema depositan en esa arquitectura.Pero esta manera de aplicación del método no permite la consideración de los riesgos asociados con actividades o eventos maliciosos durante el refinamiento de la arquitectura de supervivencia.

TRIAD puede ser extendido de una manera bastante sencilla para tratar con fallas no-maliciosas o accidentes relacionadas con la supervivencia. El modelado de dinámicas de amenazas y el análisis de impacto de fallas externas y de accidentes naturales puede realizarse de la misma manera que con los ataques maliciosos. El pronóstico preciso del impacto de las fallas internas sobre la misión puede requerir de un mayor detalle en la especificación de las operaciones internas dentro del modelo de dinámicas de amenazas. Asimismo, el modelado del árbol de ataque se puede extender con el análisis del árbol de fallas para analizar las fallas y accidentes con un menor nivel de abstracción, debido al paralelismo entre las dos técnicas.

Los riesgos no-maliciosos de programa, tales como errores de código fuente, son más difíciles de manejar con la primera de las técnicas, dado que TRIAD sólo se encarga de riesgos operativos durante la formulación de la arquitectura. Elegir la segunda técnica resulta apropiado si los riesgos de programa son altos o si se está tratando explícitamente con ellos dentro del proceso y no pueden ser pospuestos hasta después de haber formulado la arquitectura. En este caso, la integración de las actividades IAD dentro de los primeros ciclos de la espiral más comprensiva de desarrollo del sistema permite la resolución de los riesgos de programa de manera temprana, antes que muchos recursos hayan sido consumidos en un camino sin salida.

9.2.2 Trabajo futuro

TRIAD provee un sólido fundamento para los posteriores refinamiento, experimentación y validación de una solución para explotar nuestra comprensión del comportamiento de intrusos para mejorar el diseño y las operaciones de la arquitectura del sistema. Se prevén dos líneas de trabajo: ***el desarrollo de una herramienta TRIAD*** y ***la aplicación de TRIAD***.

El desarrollo de herramientas apropiadas permitirá la aplicación de TRIAD en problemas más amplios y complejos dentro de diferentes dominios. Las dinámicas de sistema proveen una base para el desarrollo de métodos y herramientas que ayuden a los ingenieros a comprender, caracterizar y comunicar el impacto de un ambiente de amenazas maliciosas sobre las operaciones de una organización y de un sistema y sus respectivas misiones. El ulterior desarrollo de dinámicas de

amenazas promete proporcionar una guía estructurada y justificable sobre cómo una organización puede adoptar las mejores políticas, procedimientos y tecnología para responder al ambiente de amenazas. La herramienta de soporte de TRIAD integrará y refinará las herramientas existentes según corresponda (por ejemplo, herramientas de dinámicas de sistema, árboles de ataque o análisis de intrusión) y dará soporte a la documentación y uso de las tácticas de supervivencia.

También se prevé explorar la viabilidad de TRIAD y su refinamiento a través de su aplicación en el análisis centrado en situaciones problemáticas muy específicas. Cada ejemplo puede implicar la identificación de una situación problemática específica, un análisis TRIAD y la mitigación de dicha situación, y una caracterización de la mejora lograda a través del análisis y mitigación. La mejora de la caracterización será una comparación de la situación problemática antes y después del análisis y mitigación TRIAD. Se planea documentar tácticas de supervivencia de una manera estructurada que facilite su comparación, composición y análisis. Los trabajos preliminares sobre este problema muestran de qué manera los patrones de ataque pueden ser estructurados de tal manera que se los puede aplicar en una variedad de contextos [Moore 01a]. Se prevé participar en los actuales trabajos llevados a cabo en el Centro de Coordinación del CERT, realizando el estudio y el análisis de los datos sobre incidentes y vulnerabilidades existentes para aprender más sobre incidentes de seguridad sobre la Internet.

Centrando el trabajo sobre un problema específico dentro de un dominio reducido, se espera lograr una rápido retorno sobre la eficacia del modelo y una mayor comprensión sobre cómo mejorarlo. El retorno que se logre ayudará a comprender las relaciones y dependencias entre las actividades y artefactos de cada sector. Se investigarán diferentes problemas y formas de mitigación en los ejemplos sobre los que se trabaje para así incrementar la experiencia ganada y la perspicacia obtenida, por ejemplo, defensas activas vs. pasivas, dominios militares vs. comerciales, soluciones COTS vs. a medida, contramedidas tecnológicas vs. de procedimiento. En cada caso, la situación problemática será restringida a una única amenaza maliciosa particular y a su impacto dentro del dominio de interés. Se espera que esta metodología fuertemente focalizada acelerará la mitigación y análisis TRIAD a una sola iteración del modelo completo, con pocos o ningún requerimiento formal de rastreo, con lo que se asegura la relativa conveniencia de los resultados.

Trabajos posteriores comprenden la aplicación a gran escala de TRIAD y de la herramienta de soporte desarrolla para demostrar su escalabilidad hacia problemas más complejos. La aplicación a gran escala requerirá del ensamblado de las actividades y estructuras TRIAD dentro de un modelo de ciclo de vida de desarrollo de sistema apropiado al dominio de aplicación y al ambiente de desarrollo. Este reporte ilustra una aproximación al desarrollo de una estrategia de supervivencia para un comercio, en el que se muestra de qué manera podría comenzarse el proceso en espiral TRIAD, seguido por iteraciones a través de actividades de sector, y su termina-

ción cuando se ha determinado un grado aceptable de riesgo residual de falla de misión. Además del refinamiento TRIAD basado en una aplicación a gran escala, se planea desarrollo un tutorial para su uso, con ejemplos relevantes, e iniciar la transferencia de la tecnología a alguna organización interesada. La herramienta de soporte TRIAD, la documentación de casos de estudio TRIAD, y una detallada guía de para la aplicación de TRIAD con configuraciones variadas debería ayudar a crear condiciones convincentes del uso y transición del modelo.

Finalmente, con una efectiva herramienta de soporte y la evidencia de su eficacia, se espera que TRIAD será integrado a modelos de ciclo de vida más generales para el desarrollo y mantenimiento de sistemas de alta confiabilidad.

9.3 CONCLUSIONES ACERCA DE LOS CONCEPTOS COMUNES EN LOS DOMINIOS DE LAS INGENIERÍAS DE PROTECCIÓN, SEGURIDAD Y SUPERVIVENCIA

Las siguientes recomendaciones se basan en las observaciones relativas a las similitudes y diferencias entre los conceptos fundamentales de las ingenierías de protección, seguridad y supervivencia.

9.3.1 Utilizar conceptos y terminología comunes

Donde resulte apropiado y práctico, utilizar conceptos y terminología comunes cuando se ejecute o describa a las ingenierías de protección, seguridad y supervivencia [Mead 03]; (por ejemplo, esta terminología común debería incluir conceptos comunes tales como activos, daño, accidente, ataque, peligro, amenaza, riesgo, vulnerabilidad, requerimientos, política, y objetivo). *Agregar conceptos nuevos para clarificar las relaciones entre términos significativamente similares* (tales como accidente, ataque o peligro y amenaza). *Definir con claridad estos conceptos de tal manera que sus similitudes y diferencias resulten obvias*. Finalmente, *codificar estas definiciones en estándares, libros y materiales de capacitación*.

9.3.2 Agregar a la Defendibilidad como un factor de calidad nuevo

Como se ilustra en la Figura 62, *crear un nuevo factor abstracto de calidad denominado defendibilidad como una sub-clase de la fiabilidad la cual, a su vez, es una sub-clase dentro de los tres factores de calidad de protección, seguridad y supervivencia. La defendibilidad puede actuar como un punto focal de conceptos comunes y procesos relacionados* (por ejemplo, el análisis de riesgo basado en activos) *que pueden ser heredados por sus tres sub-clases.*

Existe una tendencia a combinar a las ingenierías de protección y de seguridad dentro de aseguramiento de la integridad. Sin embargo, la integridad posee un significado demasiado específico

en el campo de la ingeniería de seguridad (es decir, integridad es un sub-factor de seguridad), por lo que el término "integridad" es demasiado específico y el término más general "defendibilidad" es más apropiado para la combinación de la protección, seguridad y supervivencia.

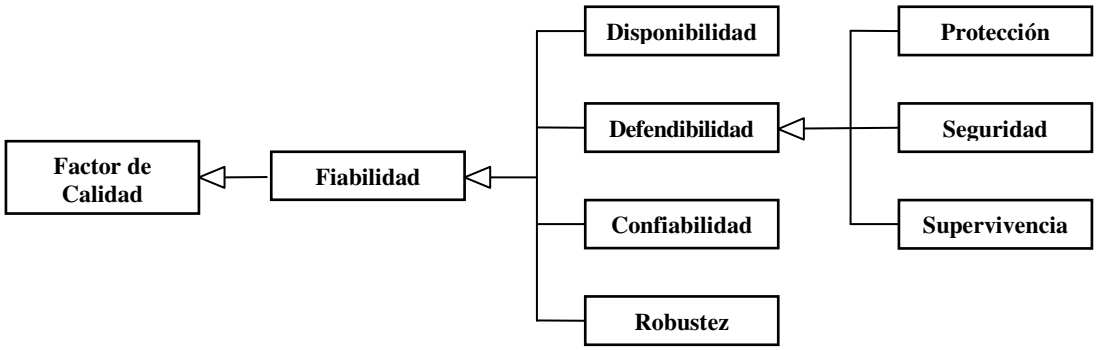


Figura 62. Defendibilidad como una clase de fiabilidad.

9.3.3 Descomponer la Defendibilidad

Descomponer la defendibilidad (y en consecuencia, la protección, la seguridad y la supervivencia) en un conjunto estándar de sub-factores de calidad: protección de activos, detección de incidentes, reacción ante a incidentes y adaptación del sistema. Estos sub-factores de calidad, a su vez, se pueden descomponer en un conjunto estándar de sub-factores de calidad de menor nivel que son esencialmente comunes a la protección, la seguridad y la supervivencia.

Con la descomposición explícita de la protección, la seguridad y la supervivencia en protección de activos, detección de incidentes, reacción ante a incidentes y adaptación del sistema se ilustra en la Figura 63, obtenemos los siguientes beneficios:

- La descomposición proporciona una deseada estandarización a lo largo de las tres clases de defendibilidades.
- A la luz del histórico énfasis sobre la protección de activos, esta descomposición ayuda a asegurar que los otros tres sub-factores no son dejados de lado y que se ha producido un conjunto completo de requerimientos de defendibilidad.
- Esta descomposición permite que las diferentes clases de defendibilidad posean sus propios sub-factores específicos del factor adicionales (por ejemplo, protección de disponibilidad, integridad, privacidad, etc., como sub-factores de protección de activos).
- Esta descomposición puede ser extendida con nuevos sub-factores a medida que vayan siendo identificados.

Los sub-factores ilustrados en la Figura 63 se pueden definir de la siguiente manera:

- **Protección de activos** (también conocido como prevención y resistencia) *es el grado*

con el cual los activos son protegidos contra accidentes y ataques. La protección de activos incluye tanto la eliminación de peligros y amenazas como así también las etapas para minimizar los efectos negativos derivados de la ocurrencia de un accidente o de un ataque exitoso. Se lo puede descomponer en varios sub-factores específicos del factor de calidad tales como control de acceso (identificación, autenticación, y autorización), protección de disponibilidad, integridad, no-repudiación, protección física y privacidad.

- **Detección de incidentes** (también conocido como reconocimiento) *es el grado con el cual accidentes y ataques relevantes (o el daño que ellos causan) son reconocidos a medida que ocurren de tal manera que el sistema puede reaccionar de manera acorde* (por ejemplo, manteniendo los servicios esenciales, degradando de manera ordenada). La detección de incidentes generalmente comprende la identificación del incidente y el *logging*. Por ejemplo, la registración de ataques puede proporcionar una evidencia legal con el cual llevar a juicio a los atacantes. También puede incluir el reconocimiento de las condiciones o eventos que preceden a los incidentes (por ejemplo, antes de la falla del hardware) o la detección de los atacantes recolectando información durante los sondeos previos a los ataques.
- **Reacción ante** incidentes (también conocido como recuperación) *es el grado con el cual el sistema responde* (por ejemplo, se recupera) *luego de un accidente o un ataque*. La reacción ante incidentes puede abarcar análisis y reportes de incidentes, degradación y restauración del sistema, como así también la acusación de las personas que causaran los accidentes y de los atacantes que montaran los ataques. La restauración del servicio generalmente implica el establecimiento de metodologías de recuperación basadas en prioridades de tal manera que cualquier servicio esencial que se pueda haber perdido o degradado sea recuperado antes que cualquier otro servicio no-esencial que se haya perdido o terminado.
- **Adaptación del sistema** *es el grado con el cual el sistema se adapta por sí mismo* (en base a los accidentes y ataques que se están produciendo) *de tal manera que en el futuro pueda proteger mejor sus activos, detectar incidentes, y reaccionar ante ellos*. La adaptación del sistema puede abarcar el análisis de las tendencias de incidentes como así también el mejoramiento de las salvaguardas y contramedidas.

9.3.4 Incluir todos los tipos de Activos

Como se ilustra en la Figura 64, *se debe asegurar que todos los activos relevantes sean considerados al momento de desarrollar los requerimientos de defendibilidad* (por ejemplo, protección, seguridad y supervivencia). Considerar cada clase de activo, no sólo las clases más conocidas

(por ejemplo, protección de las personas y seguridad de los datos). Ahora, los siguientes conceptos poseen las siguientes definiciones:

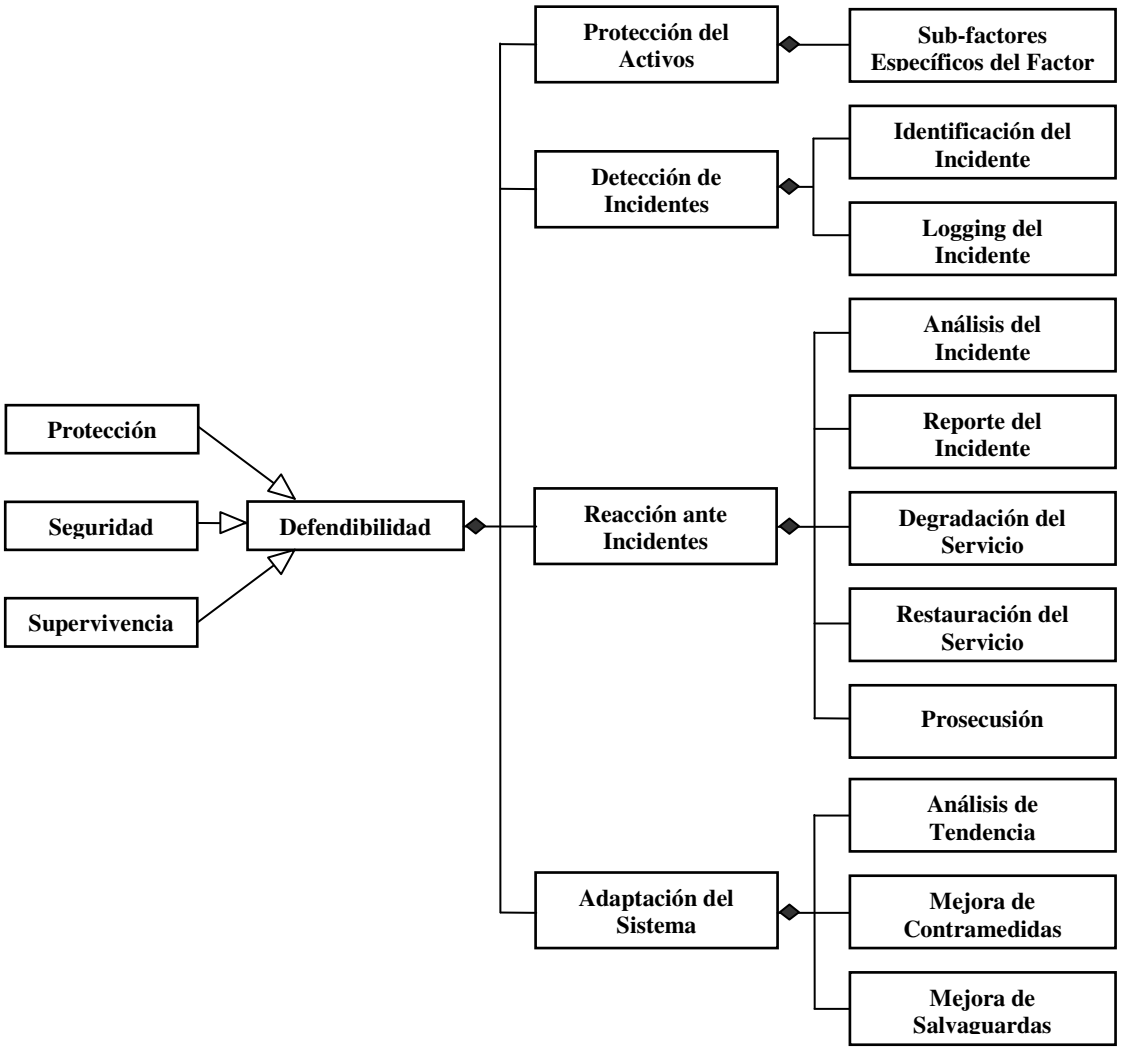


Figura 63. Descomposición estándar de la Defendibilidad en Sub-factores de Calidad.

- **Activo** *es cualquier cosa que posea un valor que deba ser protegido contra un daño*.
- **Daño** *es un deterioro grave o un impacto negativo* (es decir, un resultado negativo) *asociado con un activo debido a un incidente*. El daño se puede descomponer de acuerdo al tipo de activo dañado (por ejemplo, daño a personas incluyen cosas tales como lesiones, enfermedades, muerte o víctima de un ciber-crimen) o al tipo de incidente (por ejemplo, daño debido ataque puede incluir la exposición de información sensible). *El daño debe ser suficientemente significativo como para exigir una acción reparadora que lo evite en el futuro*.

9.3.5 Incidente

Los accidentes y ataques están obviamente relacionados en que ambos causan daños a activos, los cuales deben ser protegidos de ellos. Como se ilustra en la Figura 65, *se crea un nuevo concepto, "incidente", para que actúe como una super-clase abstracta para accidente y ataque*. Los conceptos resultantes dentro de esta jerarquía de herencias ahora poseen las siguientes definiciones:

- **Incidente** es cualquier evento o conjunto cohesivo de eventos que *pueden* causar daño a un activo de valor.

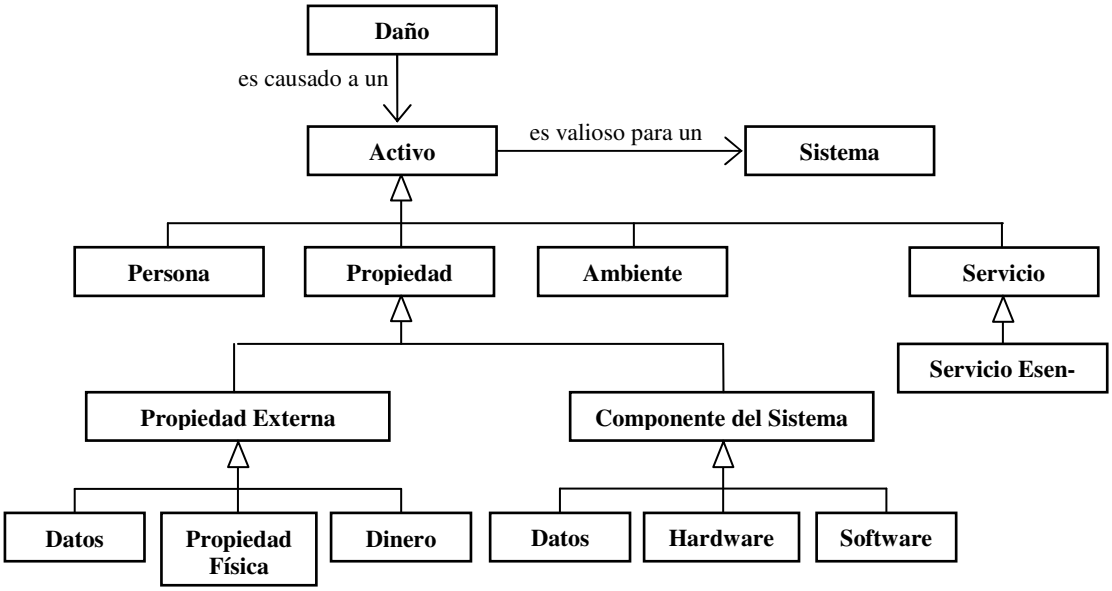


Figura 64. Activos y Daño.

  - **Incidente de protección** es cualquier incidente no planeado ni deseado, aunque no necesariamente inesperado (es decir, accidental).
  - **Accidente** (también conocido como desgracia) es cualquier incidente de protección que causa daño.
  - **"Falló por poco"** cualquier incidente de protección que no causa daño.
- **Ataque** (también conocido como brecha de seguridad) es cualquier incidente tanto intencional como no-autorizado (es decir, malicioso). En consecuencia, un ataque es un intento malicioso montado por un atacante para violar la seguridad de un sistema, evadir los mecanismos de seguridad del sistema, y causar daño a un activo.
  - **Ataque exitoso** es cualquier ataque que causa daño a un activo.
  - **Ataque no-exitoso** es cualquier ataque que no causa daño.

Como se ilustra en la Figura 65, la protección y la seguridad poseen sub-árboles de generalización

similares, en los que "incidente de protección" se corresponde con "ataque", "accidente" se corresponde con "ataque exitoso", "falló por poco" se corresponde con "ataque no-exitoso". También se debe observar a partir de las definiciones anteriores que todos los eventos asociados con incidentes no son necesariamente parte del incidente. Por ejemplo, generalmente ocurren una serie de eventos que mueven al sistema a un estado peligroso (más correctamente a un conjunto de estados incompatibles) antes de la ocurrencia de un accidente. De manera similar, a menudo una serie de eventos exploratorios o sondeos preceden y conducen a un ataque real [Allen 01].

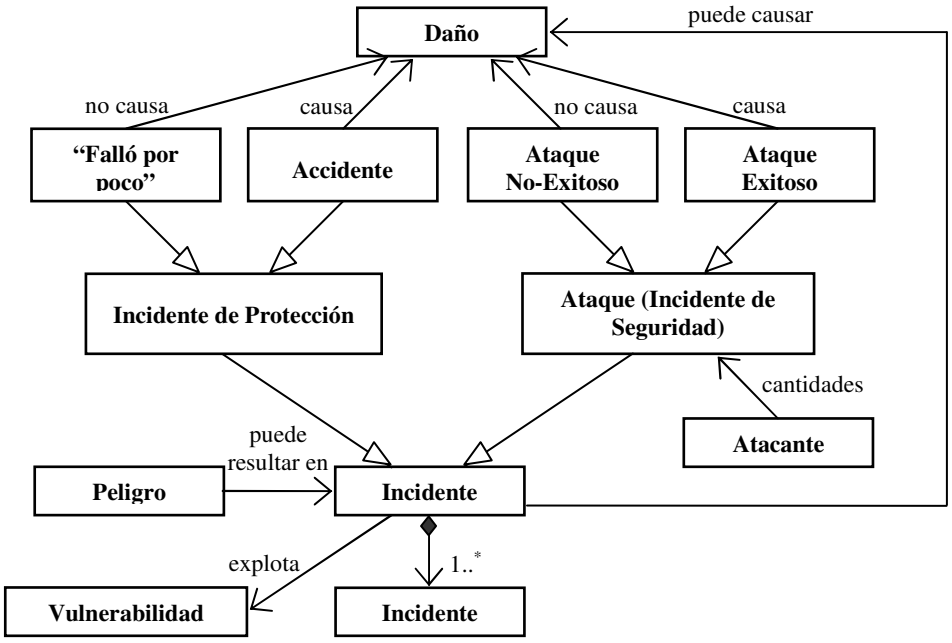


Figura 65. Incidentes (Accidentes y Ataques).

9.3.6 Peligrosidades

Al igual que con los accidentes y los ataques, también los peligros y las amenazas pueden obviamente estar relacionados con aquellos incidentes que causan daño a los activos de valor. Se ha creado un nuevo concepto, peligrosidad, que actúa como una super-clase abstracta para tanto el peligro como la amenaza. Como se ilustra en la Figura 66, estos conceptos ahora poseen las siguientes definiciones:

- **Peligrosidad** (también conocido como impedimento) es una situación (por ejemplo, un conjunto de uno o más condiciones o estados incompatibles del sistema, que posiblemente incluya una o más condiciones dentro del ambiente del sistema) que incrementa la probabilidad de uno o más incidentes relacionados.
  - **Peligro** es una peligrosidad que puede resultar en uno o más accidentes relacionados.
  - **Amenaza** es una peligrosidad que puede resultar en uno o más ataques relacionados.

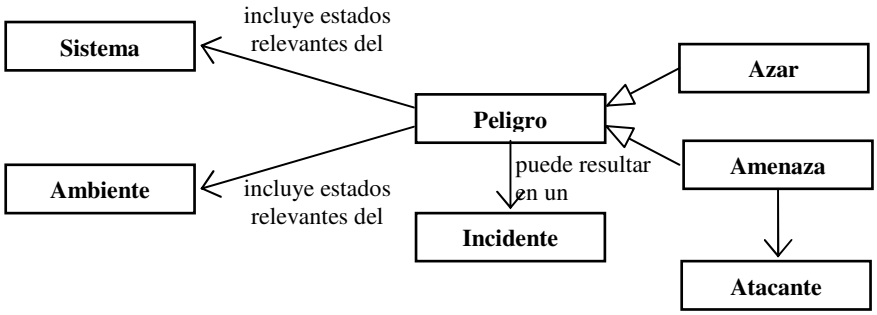


Figura 66. Peligrosidades (Peligros y Amenazas).

9.3.7 Aprovechar las características comunes de los Modelos de Información

Como se ilustra en la Figura 67, *las características comunes existentes en los diagramas precedentes se pueden combinar para así conformar un único modelo de información para la defendibilidad. Se puede reducir el esfuerzo aprovechando lo que existe en común entre los modelos de protección, seguridad y supervivencia cuando se debe identificar y analizar*:

- *Los activos que se deben proteger de un daño*
- *El daño que pueden sufrir estos activos*
- *Los incidentes (es decir, accidentes y ataques) que pueden causar el daño*
- *Los riesgos de los activos*
- *Las vulnerabilidades de los activos*
- *Los mecanismos que protegerán los activos y que satisfacen los requerimientos*

9.3.8 Desarrollo de un Proceso común

Las similitudes de los modelos de información de las Figuras 56, 57 y 58 proporcionan la justificación de la recomendación expresada en muchos libros y artículos respecto a que los requerimientos de la protección y la seguridad sean diseñados en base a la asociación entre los riesgos de peligrosidades con los activos [Alberts 03, Firesmith 03c, Herrmann 99, Moffett 03, Peltier 01]. Si los requerimientos son demasiado fuertes para los riesgos, entonces se han de gastar dinero y tiempo de manera excesiva en los componentes arquitectónicos que resultan más poderosos de lo necesario; si los requerimientos son demasiado débiles para los riesgos, entonces las peligrosidades no serán adecuadamente prevenidas, detectadas y/o se reaccionará frente a ellas. Además, *debido a que los tres tipos de defendibilidad comprenden en gran medida los mismos activos y sus peligrosidades y riesgos asociados, adquiere sentido diseñar estos factores de calidad en forma simultánea como un grupo para evitar el desperdicio de esfuerzos redundantes*. En consecuencia, *el uso de las características en común de estos modelos de información con un fundamento sobre el cual construir una metodología común y única de análisis de los riesgos de peligrosidad (peligro/amenaza) en base a activos para la ingeniería de la protección, la seguridad y la superviven-*

cia. Los siguientes pasos constituyen un ejemplo de tales procesos cuando se lo ejecutar de manera iterativa, incremental y en paralelo con las otras actividades y tareas:

1. Integrar la ingeniería de defendibilidad con el resto del proceso de ingeniería.
2. Desarrollar un programa de defendibilidad que incluya protección, seguridad y supervivencia.
3. Identificar y priorizar los activos que se encuentra sujetos a algún tipo de peligrosidad y que puede, en consecuencia, ser dañados.
4. Establecer los objetivos y políticas de defendibilidad para proteger estos activos.
5. Determinar los impactos negativos que podrían ocurrirle a estos activos en caso que las peligrosidades pudieran causar incidentes (accidentes y ataques).
6. Identificar y describir el perfil de los potenciales atacantes como así también las causas externas de los accidentes (incendios, inundaciones, etc.).
7. Identificar, categorizar y priorizar las peligrosidades (amenazas y peligros) que pueden dañar estos activos. Identificar y analizar sus potenciales causas.
8. Estimar los riesgos asociados con estos activos y priorizarlos en base a la extensión del impacto negativo que puede ocurrir y la probabilidad de la ocurrencia de la peligrosidad.
9. En orden a la prioridad del riesgo, seleccionar los sub-factores de calidad relevantes de entre los sub-factores de calidad de protección, detección y reacción correspondientes a los factores de calidad de protección, seguridad y/o supervivencia. Por ejemplo, seleccionar la autenticación o la integridad de datos con se trabaja con la seguridad.
10. Determinar uno o más criterios de calidad específicos del sistema para determinar la existencia de los sub-factores de calidad asociados.
11. Seleccionar la métrica de calidad asociada para cada criterio y determinar un nivel mínimo requerido para esa métrica de calidad.
12. Identificar, analizar y especificar los requerimientos de defendibilidad como combinaciones de criterios de calidad con un nivel mínimo de la métrica de calidad asociada. Encontrar el punto de equilibrio entre estos requerimientos y los demás requerimientos potencialmente conflictivos.
13. Definir los mecanismos (salvaguardas y contramedidas) para satisfacer estos requerimientos
14. Diseñar e implementar estos mecanismos arquitectónicos
15. Identificar y analizar cualquier vulnerabilidad remanente.
16. Realizar la verificación (por ejemplo, un testeo de seguridad)
17. Obtener la certificación y/o acreditación
18. Guardar evidencia de las acciones de defendibilidad
19. Analizar y registrar incidentes

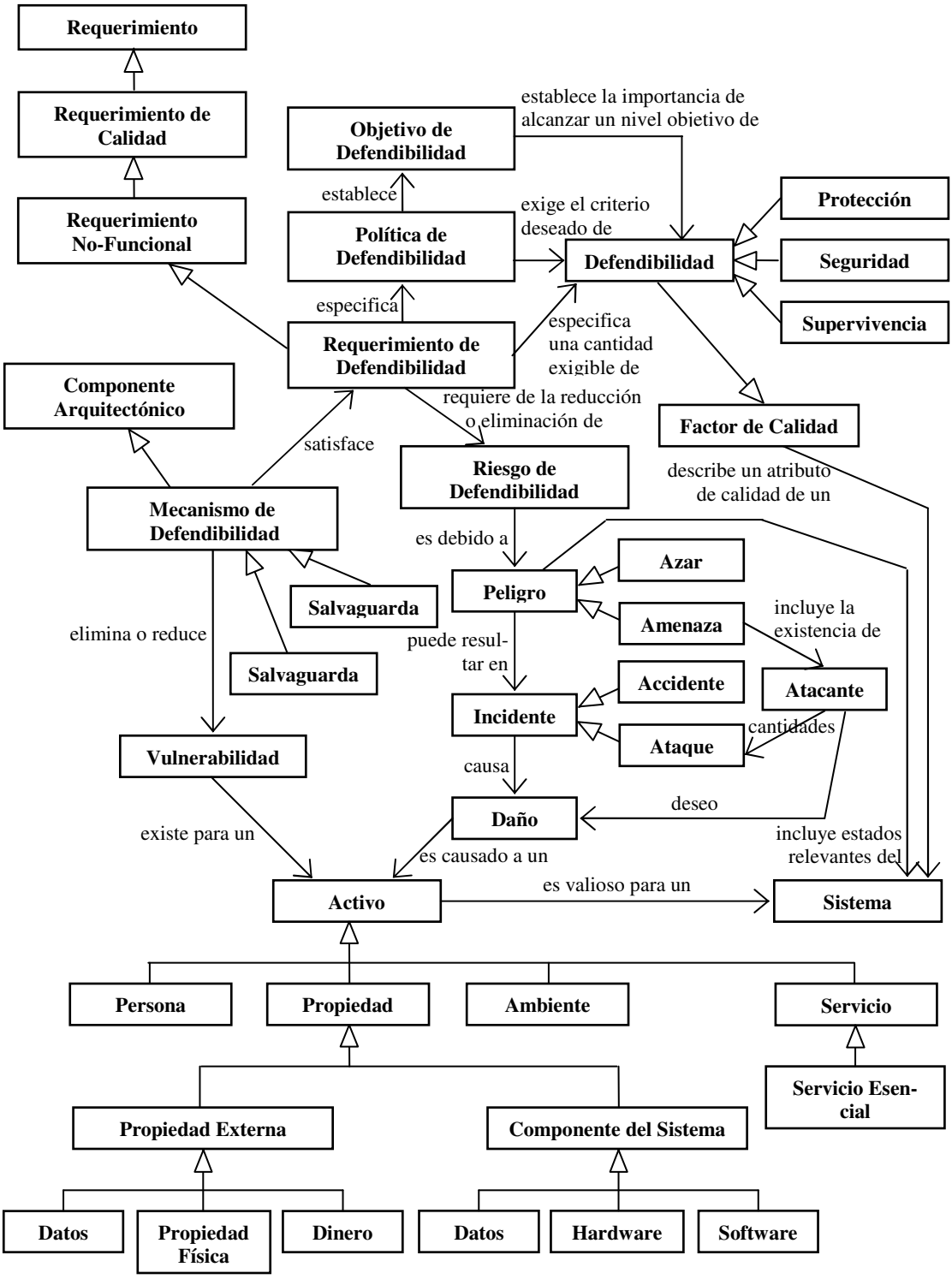


Figura 67. Modelo de Información para la Ingeniería de la Defendibilidad.

9.3.9 Disponer los Requerimientos

Incluir los requerimientos de defendibilidad resultantes en la parte apropiada de la sección de requerimientos de calidad de la especificación de requerimientos del sistema, y no en tres documentos separados que podrían ser usados o no como entrada para la arquitectura y el diseño

del sistema; por ejemplo, no confiarse en que el documento de política de seguridad contenga los requerimientos de seguridad; los requerimientos no son política, y ubicar los requerimientos de seguridad junto con las políticas de seguridad reduce la probabilidad de que sean contemplados en la arquitectura y que se testee su consistencia con los demás requerimientos.

9.3.10 Diseñar de manera temprana los Requerimientos y la Arquitectura de Defendibilidad

No se debe esperar hasta que exista el resto de la arquitectura y hayan sido determinados los componentes antes de proceder al diseño de los requerimientos de protección, seguridad y supervivencia y la selección de sus mecanismos arquitectónicos asociados. Para ese momento, los demás requerimientos se encontrarán en su mayor parte completados, y los requerimientos de defendibilidad resultantes yo no incidirán en la arquitectura y no serán chequeados en cuanto a su consistencia con los demás requerimientos. Resulta dificultoso, costoso y consume tiempo tratar de agregar protección, seguridad y supervivencia a una arquitectura existente.

9.3.11 Trabajo Futuro

Clarificar la terminología atendiendo a la protección, seguridad y supervivencia resulta particularmente importante debido a que proporciona un fundamento sólido para la ingeniería de protección, seguridad y supervivencia, es insuficiente en sí misma. Además debe proporcionar un fundamento sólido sobre el cual llevar adelante la investigación concerniente a estas tres disciplinas de la ingeniería, especialmente si uno está tratando de aprovechar estas características comunes.

El trabajo "*Common Concepts Underlying Safety, Security, and Survivability Engineering"* realizado por D.G. Firesmith constituye el basamento de un esfuerzo de investigación y desarrollo interno del *Software Engineering Institute* tendiente a ***identificar maneras de reutilizar requerimientos de protección, seguridad y supervivencia como así también los procesos recomendados para concebir y analizar tales requerimientos***.

Futuras líneas de trabajo incluyen temas tales como:

- Una mejor identificación de los sub-factores de calidad para la seguridad y la supervivencia al mismo nivel de descomposición existente para la seguridad.
- Determinar la mejor manera en que estos sub-factores de calidad se puedan fijar para prevenir, detectar y reaccionar, dado que los mismos factores de calidad parecieran acomodarse bajo más de una de estas categorías.
- Realizar investigaciones ulteriores para determinar la mejor manera de atender a la disponibilidad (normal vs. pérdida accidental vs. pérdida maliciosa vía ataques DoS) a fines de minimizar el solapamiento entre los factores de calidad.
- Determinar el solapamiento de los componentes arquitectónicos de defendibilidad.

9.4 CONCLUSIONES FINALES

La mayoría de los ingenieros en requerimientos están pobremente entrenados para elicitar, analizar y especificar requerimientos de defendibilidad, confundiéndolos muchas veces con mecanismos arquitectónicos de seguridad que tradicionalmente han sido utilizados para satisfacer requerimientos de este tipo. Por ello, terminan especificando una arquitectura o restricciones de diseño más que verdaderos requerimientos. Los reportes analizados en esta Tesis están orientados a salvar estas deficiencias de una manera abarcativa en lo que hace a la diversidad de visiones y metodologías consideradas, y de fuerte orientación formal, a fin de enriquecer la actual manera de considerar el objetivo de calidad de un sistema de información.

Hoy en día, pensar en satisfacer plenamente este objetivo, obliga necesariamente a encarar aspectos relativos con la ingeniería de requerimientos para sistemas con capacidad de supervivencia o defendibles. Y esto a su vez, implica atender a los siguientes requerimientos:

- Requerimientos de identificación: cualquier requerimiento de seguridad que especifica la manera en que una regla de negocio, una aplicación, un componente o un centro deberá identificar sus entidades externas (por ejemplo, actores humanos o aplicaciones externas) antes de interactuar con ellas.
- Requerimientos de autenticación: cualquier requerimiento de seguridad que especifica la manera en que una regla de negocio, una aplicación, un componente o un centro deberá verificar la identidad de sus entidades externas (por ejemplo, actores humanos o aplicacones externas) antes de interactuar con ellas.
- Requerimientos de autorización: cualquier requerimiento de seguridad que especifica el acceso y los privilegios de utilización de los usuarios y aplicaciones clientes autenticadas.
- Requerimientos de inmunidad: cualquier requerimiento de seguridad que especifique la manera en que una aplicación o un componente se ha de proteger a sí mismo de la infección de programas no-deseables no-autorizados (por ejemplo, virus informáticos, gusanos y troyanos).
- Requerimientos de integridad: cualquier requerimiento de seguridad que especifique la manera en que una aplicación o un componente deberá asegurar que sus datos y sus comunicaciones no son intencionalmente interrumpidas vía la creación, modificación o borrado no-autorizado.
- Requerimientos de detección de intrusión: cualquier requerimiento de seguridad que especifique la manera en la que una aplicación o un componente deberá detectar y registrar el intento de acceso o modificación por parte de individuos no-autorizados.

- Requerimientos de no-repudiación: cualquier requerimiento de seguridad que especifique la manera en que una regla de negocio, aplicación o componente deberá evitar que un participante en una de sus interacciones (por ejemplo, mensaje, transacción) niegue haber participado en toda o parte de la interacción.
- Requerimientos de privacidad: cualquier requerimiento de seguridad que especifique la manera en que una regla de negocio, una aplicación, un componente o un centro deberá proteger sus datos sensibles y sus comunicaciones privadas de individuos y programas no-autorizados.
- Requerimientos de auditoría de la seguridad: cualquier requerimiento de seguridad que especifique la manera en que una regla de negocio, una aplicación, un componente o un centro deberá habilitar al personal de seguridad a auditar el estado y el uso de sus mecanismos de seguridad.
- Requerimientos de supervivencia: cualquier requerimiento de seguridad que especifique la manera en que una aplicación o un centro deberá sobrevivir a la pérdida o destrucción intencional de un componente.
- Requerimientos de protección física: cualquier requerimiento de seguridad que especifique la manera en que una aplicación o un centro se deberá proteger a sí mismo de un ataque físico.
- Requerimientos de seguridad en el mantenimiento del sistema: cualquier requerimiento de seguridad que especifique la manera en que una aplicación , un componente o un centro deberá evitar las modificaciones autorizadas (por ejemplo, aplicar parches, aplicar actualizaciones) sin afectar accidentalmente sus mecanismos de seguridad.

Por otra parte, la especificación de buenos requerimientos exige que éstos posean varias propiedades fundamentales, tales como que sean consistentes, necesarios, no-ambiguos y completos. Atendiendo a esta última propiedad, resulta imprescindible considerar la inclusión de modelos de requerimientos, y dentro de éstos:

- Modelos de Protección: un modelo de protección es un modelo que documenta aspectos de protección del sistema y que es utilizado por el equipo de especialistas en esta área para identificar y analizar los requerimientos de protección. Durante el proceso de análisis, el equipo de especialistas puede fracasar en la creación de modelos útiles y factibles en todos los niveles de la jerarquía del sistema. Resulta fundamental verificar la inclusión y su correcta definición de los siguientes modelos:
    - Modelos de Valorización
    - Modelos de Accidente
    - Modelos de Peligro

  - Modelo de Riesgo de la Protección
- Modelos de Seguridad: un modelo de seguridad es un modelo que documenta aspectos de seguridad del sistema y que es utilizado por el equipo de especialistas en esta área para identificar y analizar los requerimientos de protección. Durante el proceso de análisis, el equipo de especialistas puede fracasar en la creación de modelos útiles y factibles en todos los niveles de la jerarquía del sistema. Resulta fundamental verificar la inclusión y su correcta definición de los siguientes modelos:
  - Modelos de Valorización
  - Modelos de Atacante
  - Modelos de Ataque
  - Modelo de Amenaza
  - Modelo de Riesgo de la Seguridad
- Modelos de Supervivencia: un modelo de supervivencia es un modelo que documenta aspectos de supervivencia del sistema y que es utilizado por el equipo de especialistas en esta área para identificar y analizar los requerimientos de protección. Durante el proceso de análisis, el equipo de especialistas puede fracasar en la creación de modelos útiles y factibles en todos los niveles de la jerarquía del sistema.

SECCION 10: UNA NUEVA VISION

En los últimos, se han venido produciendo profundos cambios en le carácter de los problemas de seguridad, en sus contextos técnicos y de negocio, y en los objetivos y propósitos de sus responsables. Como una consecuencia de ello, muchos de los supuestos subyacentes en las tecnologías de seguridad tradicionales ya no son válidos. El fracaso en reconocer la profundidad y variedad de los mismos en forma combinada frena la aplicación de soluciones efectivas a los actuales problemas de seguridad. La capacidad de supervivencia proporciona una nueva perspectiva técnica y de negocio sobre la seguridad, lo que resulta esencial para la investigación de soluciones. Más aún, la visión que se ha tenido en las recientes investigaciones analizadas en el presente trabajo expande la visión de la seguridad desde una especialización técnica "limitada", accesible sólo a los expertos de seguridad, hacia una perspectiva de gestión de riesgo que requiere de la participación de una organización como un todo (gestión ejecutiva, expertos en seguridad, expertos en el dominio del negocio, y otros responsables) para proteger de los ciberataques, fallas y accidentes a los sistemas de misión crítica.

10.1 La capacidad de supervivencia desde una perspectiva técnica

La supervivencia ha sido definida como la capacidad de un sistema de satisfacer su misión, de manera oportuna, en presencia de ataques, fallas, o accidentes, en donde el término "sistema" se utiliza en el sentido más amplio, e incluye redes y "sistemas de sistemas" de gran escala. Si bien esta definición captura el concepto que se ha venido desarrollando de supervivencia de una manera suscinta, no expone de manera clara el raciocinio y las implicancias de esta visión.

En tanto la seguridad tradicionalmente se ha centrado en la confidencialidad de la información, los problemas que más preocupan hoy en día se relacionan con la disponibilidad de la información y la continuidad de los servicios. La preocupación en cuanto a la continuidad de servicios críticos que manifiestan los proveedores de servicios, sus clientes, y los organismos gubernamentales ha conducido a la creación de un nuevo campo en el aseguramiento de las infraestructuras. La viabilidad comercial de las empresas depende de su habilidad de producir y entregar sus productos y servicios de manera oportuna. Estos son objetivos de la misión que van más allá, y por lo tanto deben ser extendidos, del alcance tradicional de la seguridad.

La mayor parte de la tecnología de seguridad depende de un conjunto de supuestos subyacentes acerca de la naturaleza y estructura de los sistemas. Generalmente, estos supuestos incluyen la idea de que los sistemas son cerrados, que se encuentran bajo un control administrativo centralizado (o unificado), y que los administradores tiene la capacidad de observar cualquier actividad que se produzca dentro del sistema. Estos supuestos pueden haber sido apropiados cuando los

sistemas eran islas con interfaces altamente controladas hacia el resto del mundo. Sin embargo, en la actualidad, los sistemas son abiertos, y ninguna persona u organización tiene el control administrativo total. Los observadores, sean que se encuentren dentro o fuera del sistema, sólo tienen una visibilidad limitada de la estructura, extensión o topología del sistema. La falta de un control administrativo centralizado y la ausencia de visibilidad son las propiedades de la Internet, de cualquier aplicación residente en la Internet y de cualquier infraestructra dentro de una industria desregulada. Al prevalecer hoy día estos sistema ilimitados (unbounded) sin ningún control centralizado ni visibilidad global, resultan incompatibles con los supuestos que sustentan las actuales tecnologías de seguridad existentes.

La capacidad de supervivencia a menudo implica establecer un punto de equilibrio entre varios atributos de software y de hardware. Esto a llevado a que muchas personas lleguen a la incorrecta conclusión de que la capacidad de supervivencia es sinónimo de fiabilidad (dependability, en idioma inglés). Este equilibrio en lo que hace a dicha capacidad se debe establecer entre los requerimientos funcionales y no-funcionales determinados por su misión. Los atributos de calidad de software críticos para una misión o una aplicación pueden ser irrelevantes para otra. Además, la metodología tradicional de alcanzar la fiabilidad (y la seguridad) es asegurando ciertos atributos de calidad en los componentes de un sistema, y luego valerse de ellos durante el proceso de composición que preservará esas cualidad en el sistema como un todo. Pero para la supervivencia y la safety, en general, no existe tal proceso de composición. De hecho, un supuesto fundamental de la capacidad de supervivencia es que ningún componente es inmune al compromiso, accidente, o falla. En cambio, las propiedades globales funcionales y no-funcionales de un sistema que posee esta capacidad debe emerger en virtud del proceso de composición a partir de componentes que no poseen capacidad de supervivencia. Por lo tanto, debe reconocerse que la supervivencia es un propiedad emergente que no se puede alcanzar en un nivel de componentes atómicos del sistema, debido a que cada componente representa un único punto de falla para su propia supervivencia. Un ejemplo bien conocido del concepto de propiedades emergentes es la creación de sistemas estables a partir de componentes que son menos estables que el sistema compuesto. Esto presenta cierta analogía con la creación de sistemas con capacidad de supervivencia a partir de componentes sin dicha capacidad.

La mayor parte de las tecnologías de seguridad derivan de un modelo de fortaleza en el cual existe una clara distinción entre las personas de confianza y los demás potenciales usuarios e intrusos. En las aplicaciones altamente distribuidas y los sistemas basados en Internet actuales existe una escasa distinción entre los “de adentro” y los “de afuera”. Cualquier que elija conectarse a la Internet es uno “de adentro”, sea o no conocido para un subsistema particular. Esta característica se deriva de las necesidades actuales de conectividad. Las organizaciones no pue-

den sobrevivir en ambientes altamente competitivos sin ofrecerles a sus clientes, proveedores y socios un acceso fácil y rápido. Más y más, los socios en un proyecto son los competidores del siguiente, de tal manera que la confianza se torna un concepto extremadamente complejo. Las relaciones de confianza están cambiando continuamente, y en los términos tradicionales pueden ser altamente ambiguas. La confianza resulta especialmente dificultosa de establecer en presencia de usuarios desconocidos que aparecen desde fuentes desconocidas que están por fuera del propio control administrativo.

Un modelo de fortaleza es tan robusto como su componente más débil. Si uno "de adentro" de confianza abusa de su autoridad, o un intruso descubre una vulnerabilidad explotable dentro de un perímetro de seguridad, puede estar comprometido todo el sistema. En las redes ilimitadas en las que todos son "de adentro" y a menudo desconocidos, siempre existen muchos que no son dignos de confianza. Más aún, los modelos de fortaleza no permiten un *graceful degradation*[38], ni los mecanismos *failsafe*[39] demandados por los objetivos de disponibilidad y de la misión.

Por lo tanto, las características diferenciadoras entre la capacidad de supervivencia y la seguridad tradicional son propósitos y objetivos, el contexto técnico, el contexto de negocio, las restricciones técnicas, los supuestos subyacentes respecto de la tecnología aplicable, y la efectividad potencial de las tecnologías particulares. No debe sorprender demasiado que, en combinación, el propósito, el contexto, las partes implicadas, y los supuestos de las tecnologías de seguridad existentes resulten menos que satisfactorios.

No es asombroso, entonces, que los avances en las herramientas, métodos y prácticas de la seguridad en los años recientes han estado dominados por los intentos de modificar y adaptar los sistemas modernos para que se ajusten a los supuestos de la tecnología de seguridad tradicional. En consecuencia, a pesar de la proliferación de los sistemas abiertos e ilimitados en los que existe escasa confianza, los firewalls continúan siendo el principal mecanismo de supervivencia. A pesar de la amplia evidencia de que muchísimas vulnerabilidades (tanto conocidas como por conocer) pueblan los COTS, y el hecho de que estos productos están ampliamente disponibles para ser analizados por los potenciales atacantes en busca de debilidades explotables, se sigue estando cada vez más comprometidos con tales soluciones. De muy buena gana se incorporan software COTS como componentes de grandes sistemas, los cuales luego pagan el precio de sufrir ataques basados en las vulnerabilidades de dichos componentes. A pesar de que la diver-

[38] También *fault-tolerance*. Es la propiedad que permite que un sistema continúe operando apropiadamente ante el evento de la falla de alguno de sus componentes. Si su calidad operacional decrece por completo, dicha reducción es proporcional a la severidad de la falla, en comparación con un sistema diseñado de manera ingenua que aún una pequeña falla puede provocar el colapso total. Dicha propiedad no es sólo una propiedad de un sistema individual, sino también caracteriza la manera en que éste interactúa.

[39] Protección automática de programas y/o sistemas de procesamiento en los que una falla de hardware o de software es detecta dentro de un sistema computacional.

sidad es el mecanismo más efectivo en lo que hace a seguridad y capacidad de supervivencia en los sistemas en red actuales, se continúan desplegando idénticas implementaciones de estándares frágiles de un mismo fabricante. Se adquiere software basado en característica y costo inicial, en lugar de hacerlo en base a su robustez o costo a largo plazo o indirecto.

Lo que es peor aún, mientras se fracase al intentar reconocer y aceptar explícitamente el hecho que estos cambios combinados en el propósito, el contexto y las restricciones requieren del establecimiento de nuevos supuestos a partir de los cuales construir nuevas clases de soluciones, la tecnología continuará estando en conflicto con metodologías efectivas de solución. Como una consecuencia de todo esto, resultará imposible explotar las propiedades del nuevo dominio del problema de cara a soluciones radicales y no anticipadas.

Pero existe confianza y cierta evidencia de que pueden surgir soluciones para los problemas de supervivencia en redes ilimitadas acompañadas con avances en la diversidad, robustez, adaptabilidad y soluciones algorítmicas (frecuentemente denominados **algoritmos emergentes**[40]) que generen propiedades globales no-funcionales predectibles a partir de interacciones locales simples. Uno de los puntos centrales que este trabajo ha sido verificar que el foco de la seguridad se ha movido lo suficiente en varias dimensiones como para justificar la existencia de nuevos supuestos fundamentales. Por otra parte, reconocer que frente a la ausencia de estos nuevos supuestos la comunidad ha venido fracasando en el intento de explotar aquéllas características únicas resultantes de la revisión del espacio del problema con paradigmas compatibles, más que conflictivos. La capacidad de supervivencia proporciona una nueva perspectiva técnica y de negocio acerca de la seguridad que puede guiar hacia una mejor comprensión de la naturaleza y la estructura de los sistemas modernos altamente distribuidos, y que permitan abordar soluciones para los problemas de seguridad que hoy aparecen como inabordables.

10.2 La capacidad de supervivencia desde una perspectiva de las organizaciones

Muchas organizaciones poseen planes de contingencia para hacer frente a las interrupciones del negocio causadas por desastres naturales o accidentes. Si bien la mayoría de los ciberataques son de naturaleza menor, un ataque de este tipo sobre un sistemas de información en red de una organización crítica tiene la potencialidad de causar perturbaciones severas y prolongadas, ya sea que la organización haya sido atacada específicamente o si es una víctima aleatoria de un ataque difundido en forma general. Si un ciberataque perturba las funciones de una organización e interrumpe los servicios esenciales en los que dependen los clientes, entonces la supervivencia

[40] En una interpretación informal, como una computación distribuida eficiente que genera y preserva aquellas propiedades globales del sistema que constituyen los requerimientos derivados de la misión de un sistema. Estas propiedades globales son tanto funcionales como no-funcionales y se las denomina "propiedades emergentes" debido a que generalmente emergen de las acciones e interacciones de los diferentes componentes de una sistema combinadas, y que a menudo no prevalecen, o no pueden hacerlo, dentro de los componentes individuales del sistema.

de la misma organización se encuentra en riesgo[41].

La capacidad de supervivencia es una disciplina emergente que combina la seguridad de la infraestructura TIC con la gestión de riesgo, con el propósito de proteger los servicios y los activos altamente distribuidos. Un supuesto fundamental es que no existe ningún sistema totalmente inmune a los ataques, accidentes o fallas. En consecuencia, el foco de esta nueva disciplina no son sólo los intrusos informáticos, sino también asegurar que las funciones de misión crítica están sustentadas y que se entrega el conjunto de servicios esenciales, aún en presencia de ciberataques. El mejoramiento de la capacidad de supervivencia bajo estas condiciones mejora la capacidad de sobrevivir a accidentes y fallas del sistema que no son de naturaleza maliciosa. La seguridad informática tradicional es una disciplina altamente especializada que intenta hacer fracasar a los intrusos por medios técnicos que son fuertemente dependientes del dominio de la aplicación o del sistema que están siendo protegidos. Los firewalls, la criptografía, el control de acceso, la autenticación, y otros mecanismos utilizados en la seguridad informática son medios para proteger la aplicación subyacente de maneras muy similares sin considerar la aplicación específica que está siendo protegida. En contraste, la capacidad de supervivencia focaliza la misión de una manera mucho más aguda, y está más próxima a la gestión de riesgo que al estudio de cualquier aspecto técnico de la ingeniería de software, incluidos los atributos de calidad del software particular o compuesto. En última instancia debe atender a la misión, y no a un componente particular del sistema o al sistema mismo. La misión debe seguir aún si un ataque provoca suficiente daño o incluso la destrucción del sistema que soporta la misión. De esta manera se desplaza hacia la gestión del riesgo, un aborje que está íntimamente ligado con las características que son específicas de la misión de la aplicación que está siendo protegida, lo que representa un cambio más radical en el paradigma a medida que esta nueva disciplina de la capacidad de supervivencia de la información continúa emergiendo.

Las soluciones de supervivencia se entienden mejor como estrategias de gestión de riesgo que dependen, en primer lugar, de un íntimo conocimiento de la misión que está siendo protegida. Focalizándose en la misión, las soluciones de supervivencia se expanden más allá de las soluciones puramente técnicas independientes, aún cuando dichas soluciones técnicas están basadas en principios que exceden el alcance de la seguridad informática tradicional, la que incluye la tolerancia a fallo, la fiabilidad, la usabilidad, entre otros. Las estrategias de mitigación de riesgo se deben crear antes que nada en el contexto de los requerimientos de la misión (los conjuntos de requerimientos normales y

[41] Una diferencia significativa entre las perturbaciones causadas por desastres naturales y aquéllas causadas por ciberataques (además de la noción de un adversario inteligente que está detrás del ciberataque) es que frente a un desastre natural el cliente espera una disminución del servicio. Pero de cara a un ciberataque, es muy probable que se lo interprete como un signo de incopetencia por parte de la organización. A menos que el ciberataque sea generalizado, ningún cliente lo encontrará de su gusto.

de stress priorizados), y se debe basar en el análisis del "qué pasa si" de escenarios de supervivencia. Sólo entonces se puede encarar la búsqueda de soluciones de software genérico basado en la seguridad informática, el análisis de otros atributos de calidad de software, u otros enfoques estrictamente técnicos para soportar las estrategias de mitigación de riesgo.

Consideremos la analogía de un productor de hortalizas con la misión de aprovisionar con alimentos a un pueblo. El productor puede tener un alambrado que impida el ingreso al sembrado de ganado, liebres y otros intrusos (seguridad tradicional). Puede contar con un sistema de riego que lo utiliza en caso de sequía (redundancia). Puede haber sembrado una variedad de hortalizas para que aún en caso de condiciones ambientales adversas (por ejemplo, pestes) afecten a alguna, las demás sobrevivirán (diversidad). Todo esto está correcto. Pero aún en el caso en que todos los cultivos fracasen y no crezca ninguna de las variedades sembradas, la misión aún puede tener éxito si el productor tiene una estrategia alternativa basada en la misión de proveer alimentos -no necesariamente con una producción obtenida del ecosistema local. Si los cultivos fracasan, el productor todavía puede pescar para satisfacer la misión de proveer de alimentos a la población que depende de él. La pesca, ¿es una estrateia de seguridad, fiabilidad o tolerancia a falla? No -la misma está por fuera del sistema de producción de alimentos. Ésta es una estrategia de gestión de riesgo que sólo puede ser formulada a partir de un entendimiento profundo de que lo que debe sobrevivir es la misión. La experticia técnica en relación a la construcción de la infraestructura de un establecimiento hortícola, e incluso de la agricultura, resulta buena pero inadecuada en comparación con las estrategias basadas en un íntimo conocimiento de los requerimientos de la misión.

La capacidad de supervivencia depende no sólo del uso selectivo de soluciones de la seguridad informática tradicional, sino también del desarrollo efectivo de estrategias de mitigación del riesgo basado en el análisis "qué pasa si" conducido por escenarios y en la planificación de contingencias. Los "escenarios de supervivencia" postulan un amplio rango de ciberataques, accidentes y fallas que ayudan en el análisis y la elaboración de dichos planes. Sin embargo, para reducir la explosión de posibilidades inherentes a la creación de conjuntos representativos de escenarios de supervivencia, éstos se deben focalizar en los efectos adversos más que en las causas. Los efectos también son de una mayor importancia situacional inmediata que las causas, debido a que son con lo que nos deberemos enfrentar (y sobrevivir!) mucho antes de poder determinar si la causa fue un ataque, un accidente o una falla. Esperar a contar con un post-morten detallado para determinar la causa antes de mitigar el efecto está fuera de toda pregunta cuando se trata de la supervivencia de las más modernas aplicaciones de misión crítica.

Los planes de contingencia (incluído el desastre) requieren de decisiones de gestión de riesgo y balances económicos que sólo puede ejecutar el nivel gerencial de la organización (preferentemente con el asesoramiento de expertos técnicos en el dominio de aplicación, en seguridad in-

formática, o disciplinas de ingeniería de software relacionadas). La capacidad de supervivencia depende como mínimo de las destrezas en lo que hace a gestión de riesgo que posee el área de gestión de la organización, y está por encima de la experticia de un equipo de expertos de seguridad informática. (Y cuando se habla de destrezas se lo plantea en el contexto de la capacidad de gestionar el riesgo en el contexto de la misión y objetivos específicos del negocio, y no desde una experticia técnica abstracta en la ciencia de la gestión de riesgo). Esto es lo que resulta apropiado desde una perspectiva organizacional, debido a que la gestión de riesgo del negocio es una función principal de la gestión ejecutiva y no el rol de expertos de seguridad informática o de algún gurú técnico. Y la experticia en la gestión de riesgo y de la misión de la organización reside en dicha gestión de la organización. El rol de los expertos en seguridad, en el dominio de la aplicación y de otras áreas técnicamente relevantes es la de proveer al nivel superior de gestión de la información necesaria para la toma de decisiones de gestión de riesgo[42]. De esta manera, los pasos preliminares necesarios para que la capacidad de supervivencia alcance a la organización como un todo, y no sólo a los expertos en seguridad.

Hacer frente a eventos adversos de alto impacto, sin esperar a tener un diagnóstico definitivo de las causas, resulta central para el paradigma de la supervivencia. El manejo exitoso de tales eventos depende mucho más de una gestión de riesgo prudente y de adecuados planes de contingencia por parte de la gestión ejecutiva que del enfoque técnico específico realizado por los expertos en seguridad y otros gurús. Por ejemplo, una solución técnica "perfecta" podría resultar económicamente inviable. La factibilidad de muchas soluciones técnicas sólo pueden ser evaluadas en el contexto global del negocio. La gestión ejecutiva, mediante su planificación de contingencia, debería considerar soluciones a nivel del negocio que pueden transcender a las soluciones puramente técnicas. Una posibilidad podría ser establecer un acuerdo con otras organizaciones que le pudieran proporcionar la capacidad de la que se carece durante un determinado período de tiempo para atender sólo a aquéllos aspectos que hacen a la misión de la organización. Otra alternativa (apelando a especialistas del derecho y no a tecnólogos) sería haber incluido en el contrato de servicio una cláusula de limitación de la responsabilidad, en la que se establece que el cliente debe afrontar los riesgos de la falta de servicio. Esto podría poner a los clientes en conocimiento de que ellos deben tener prevista su propia redundancia, mientras que la variante anterior era el proveedor del servicio quien tenía que contar con la redundancia mediante un acuerdo con una proveedor alternativo. (Debido a que al tomar conocimiento de la existencia de ciertos riesgos inherentes en la

[42] El principal rol de los expertos técnicos (en los dominios de la seguridad y de la aplicación) es el de asegurar que las soluciones que soportan las estrategias de gestión de riesgo alternativas son técnicamente adecuadas. Por ejemplo, un bote salvavidas de un barco provee un medio de flotación alternativo para el salvataje de vidas ante el evento de hundimiento del barco, pero el bote debe ser apto para la navegación en alta mar y capaz de transportar la cantidad esperada de pasajeros. De lo contrario esta estrategia de supervivencia es fatalmente deficiente. La experticia de la gestión ejecutiva en la gestión de riesgo no puede reemplazar a la experticia técnica, sino que se debe construir a parir de la misma.

entrega del servicio y de un posible incremento en el valor percibido por el servicio ininterrumpido, la alternativa legal de la limitación de la responsabilidad podría incluso generar algún interés en el cliente en solicitarle al proveedor original del servicio la provisión de redundancia por un cargo adicional). Esta salida legal no es algo con lo que tengan que lidiar los expertos técnicos, sino que una manera efectiva de asegurar la capacidad de supervivencia de la misión y objetivos del negocio. Como lo ilustra este ejemplo, el punto de vista de la gestión de riesgo implica una "economía de la supervivencia" que le permite a los negocios estar adecuadamente preparados y sobreponerse a los efectos adversos de los ciberataques, accidentes y fallas con soluciones que pueden transcender a aquéllas que ofrecen sólo los expertos técnicos.

Comparemos esta nueva perspectiva con las actuales prácticas de gestión relacionadas con la seguridad. El rol principal en la toma de decisiones del nivel superior de gestión, desde el punto de vista de la seguridad tradicional, es primordialmente determinar el presupuesto directo y otros recursos que se le deben asignar a los expertos en seguridad de la organización para atender a un propósito débilmente definido de "fortalecer la seguridad" con un nivel de práctica conforme a estándares de la industria vagamente articulados. En el pensamiento de la gestión, la vinculación percibida entre los costos de la seguridad y la misión del negocio (y el resultado neto del negocio) es poco firme.

"Si gasto más dinero en seguridad informática se reducirá mi riesgo de intrusión. Pero esto ¿reducirá los riesgos significativos de mi misión de negocio? ¿Qué riesgos se reducirán, y en cuánto?" Sin un claro beneficio visible para la gestión, los costos resultantes de la seguridad generalmente serán inadecuados de satisfacer aún para los limitados objetivos técnicos de los expertos de seguridad. En la mayoría de los casos, lo que lamentablemente se soslaya un análisis en profundidad de las amenazas sobre la misión de la organización y el correspondiente análisis costo-beneficio de las estrategias de mitigación de riesgo y planes de contingencia. Los expertos de seguridad informática, aislados del entendimiento íntimo de la misión de la organización que poseen los niveles de gestión, no están en posición de realizar el análisis de amenazas necesario, excepto desde la perspectiva parcial de sus especialidades técnicas.

Como ejemplo de ello, consideremos los programas gubernamentales que se están planteando en algunos países, tales como los EEUU y de la UE, tendientes a asegurar las infraestructuras críticas para su funcionamiento. La preocupación del gobierno por el aseguramiento de la infraestructura crítica ha ayudado a alimentar el interés en la capacidad de supervivencia, pero este interés no está siendo canalizado por los directos implicados y beneficiados, tales como los actores de las áreas de la energía, el transporte, la banca y las telecomunicaciones. El gobierno le está solicitando a la industria que participe en los programas de aseguramiento de la infraestructura crítica, con la motivación de que estos programas son de principal interés para la na-

ción, las empresas y sus clientes. Pero ninguna de estas comunidades desean hacerse cargo del incremento en los costos que esto significa. La verdadera inversión en la protección de la infraestructura crítica se ha de producir cuando los ejecutivos comprendan que estos cambios son esenciales para su competitividad y su rentabilidad.

Desafortunadamente muchas de las empresas involucradas en estos programas están exigiendo soluciones técnicas, las cuales podrían ir en contra de las necesidades de sus clientes y de su propia rentabilidad. Se requiere de un mayor conocimiento de los aspectos de gestión de riesgo del negocio en lo que hace a la supervivencia, de tal manera que las organizaciones que operan las infraestructuras críticas de dichos países puedan estar motivadas por su propio interés en asegurar su propia supervivencia. De esta manera, el aseguramiento de la infraestructura crítica podrá basarse en el balance a nivel de la gestión de riesgo, el que depende de las misiones y objetivos globales del negocio, y no solamente de parches técnicos que son independientes de dichos objetivos.

En resumen, se está produciendo un revolucionario cambio técnico en las aplicaciones de negocio desde los sistemas stand-alone y cerrados sobre los cuales las organizaciones ejercen un control completo, hacia sistemas basados en COTS, abiertos y altamente distribuidos sobre los cuales sólo son posibles un control y un conocimiento profundo muy limitados. No sólo la mayoría de los servicios vía Internet están por fuera del control de las organizaciones que hacen uso de los mismos, sino también lo están la funcionalidad y los atributos de calidad de software de los sistemas basados en COTS utilizados para la construcción de las aplicaciones de negocio. Este cambio técnico nos está llevando tan lejos que ya no se pueden resolver completamente los problemas de seguridad en el dominio de lo técnico.

Desde la perspectiva tradicional de la seguridad informática, la gestión ejecutiva nunca estuvo lo suficientemente involucrada. Los expertos de seguridad simplemente presentan una solicitud de presupuesto para gestionar soluciones técnicas genéricas, independientemente del análisis de amenazas que son específicas de la misión que está siendo protegida. La gestión ejecutiva debe estar en conocimiento de las amenazas que acechan a la misión del negocio, y debe estar íntimamente involucrada en la formulación de las estrategias de mitigación de riesgo específicas de la misión.

Más aún, los expertos técnicos necesitan estar en conocimiento de las cuestiones del negocio que conducen a las cuestiones técnicas que ellos deben encarar. Sólo así podrán contribuir efectivamente con la gestión de riesgo necesaria para asegurar la supervivencia de las aplicaciones altamente distribuidas de misión crítica, operando en dominios ilimitados, de cara a los ciberataques, los accidentes y las fallas del sistema.

10.3 Un nuevo paradigma

En los últimos años se han producido cambios de fondo en la naturaleza y estructura de los sis-

temas de información. Las soluciones tradicionales de seguridad ya no resultan suficientes para hacer frente a los modernos problemas de seguridad asociados con los sistemas altamente distribuidos de misión crítica, los cuales no tienen ni un control administrativo centralizado ni una visibilidad global. Se requiere de nuevos fundamentos y de nuevos paradigmas para proteger dichos sistemas de los ciberataques.

La supervivencia es una disciplina emergente que combina la seguridad informática con la gestión de riesgo del negocio, con el propósito de proteger los servicios y los activos de información altamente distribuidos. Un supuesto fundamental es que ningún sistema es totalmente inmune a ataques, accidentes y fallas. En consecuencia, el foco de esta nueva disciplina no es sólo combatir a los intrusos informáticos, sino también asegurar que las funciones de misión crítica están sostenidas y que los servicios esenciales están siendo entregados, a pesar de la presencia de ciberataques, fallas y accidentes. Las soluciones de supervivencia se comprenden mucho mejor como estrategias de gestión de riesgo que antes que nada dependen de un estrecho conocimiento de la misión que está siendo protegida. El foco de la misión expande las soluciones de supervivencia más allá de las soluciones técnicas puramente genéricas ("una a la medida de todos"), aún cuando dichas soluciones sean de uso masivo. Las estrategias de mitigación de riesgo primero y principal se deben crear en el contexto de los requerimientos de la misión (conjunto priorizado de requerimientos normales y de stress), y se deben basar en el análisis "qué pasa si" de escenarios de supervivencia. La capacidad de supervivencia depende tanto de la comprensión real por parte del nivel gerencial de sus objetivos como de la experticia técnica de sus expertos en seguridad. Esto es lo apropiado desde la perspectiva de una organización, debido a que la gestión del riesgo de negocio es una función principal de la gestión ejecutiva, y no el rol de los expertos de seguridad o de algún otro equipo técnico. Esto de ninguna manera significa que los problemas de supervivencia no deban ser atendidos por soluciones técnicas, sino que las soluciones requieren de una relación que puede y debe integrar consideraciones técnicas y del negocio. También significa que la capacidad de supervivencia requiere de soluciones técnicas que consideren el contexto verdadero en el que un sistema debe operar, y que cualquier solución independiente de la aplicación será inadecuada.

SECCION 11: ANEXOS

ANEXO 1 – REPORTE DE CLIENTE DEL METODO SNA

Un Reporte SNA típico posee las siguientes secciones:

- *Resumen ejecutivo*. Provee una vista rápida de las recomendaciones del Método SNA y una breve discusión del sistema bajo estudio.
- *Secciones*

  1. Reseña. Proporciona una breve introducción del proyecto.

  2. Método Survivable Network Analysis. Presenta una descripción abreviada del método.

  3. Arquitectura. Describe la arquitectura del sistema bajo estudio.

  4. Servicios esenciales. Describe los procesos de negocio de la organización, los escenarios de utilización normales, y los escenarios de servicios y componentes esenciales.

  5. Escenarios de intrusión. Describen los perfiles generales del atacante, los impactos específicos del ataque sobre el sistema, los perfiles específicos del atacante del sistema, los escenarios de intrusión, los patrones de ataque, las intrusiones representativas, los tipos de atacantes, y el mapeo de los escenarios de intrusión con los servicios esenciales.

  6. Recomendaciones. Las áreas de política y arquitectura, e incluye el Mapa de Supervivencia.

  7. Implementación. Se discuten las consideraciones de la línea de tiempo y recursos.
- *Apéndices y referencias*

La descripción Método del SNA que aparece como **Resumen Ejecutivo** al principio del Reporte generalmente se lo emplea abreviado en la **Sección 2 Método SNA**. En la **Sección 3 Arquitectura** se ha previsto un Diagrama de la Arquitectura; en la Figura 68 se muestra un ejemplo de este Diagrama.

Sección 4 Servicios esenciales

Se ha previsto un mapeo de los procesos de negocio y los servicios esenciales; en la Tabla 16 se muestra una plantilla para el mismo.

Procesos de negocio / **Servicios/Activos esenciales**	**Proceso 1**	**Proceso 2**	**Proceso 3**
Servicio1 / Activo1 Esenciales	**X**		
Servicio2 / Activo2 Esenciales	**X**	**X**	
Servicio3 / Activo3 Esenciales	**X**	**X**	**X**
Servicio4 / Activo4 Esenciales		**X**	**X**

Tabla 16. Mapeo de procesos de negocio y servicios esenciales.

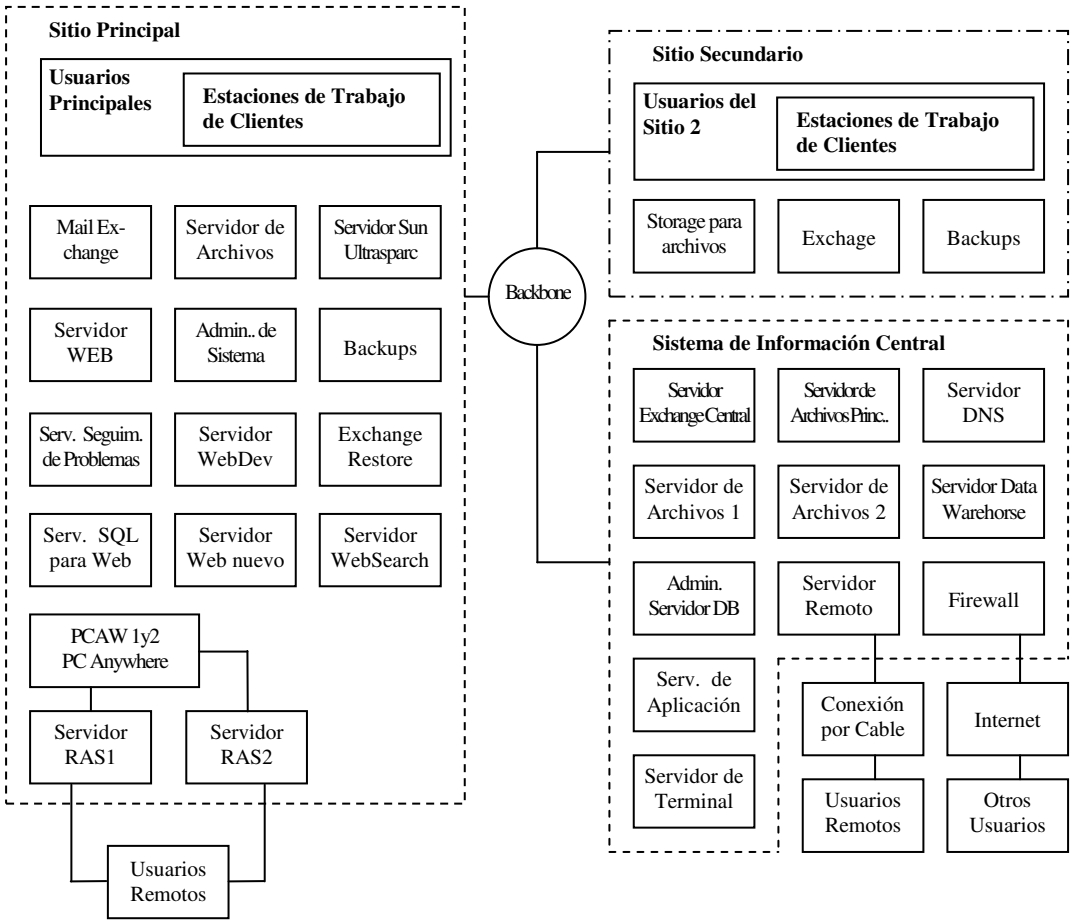


Figura 68. Ejemplo del Diagrama de Arquitectura.

El Diagrama de Arquitectura está modificado para mostrar los componentes de servicios en negrita, para cada uno de los escenarios de servicios esenciales. Un único diagrama muestra la unión entre estos diagramas, ofreciendo una vista rápida de todos los componentes esenciales; en mismo se ilustra en la Figura 69.

Sección 5 Escenarios de intrusión

A. Atributos del atacante

En esta sección se identifican las clases de atacantes; para cada clase de atacantes, se han destacado los siguientes atributos:

- **Recursos** Los recursos a los que un atacante puede recurrir; incluyen fondos, personal y niveles de destreza que este personal.
- **Tiempo** Un atacante puede tener objetivos muy inmediatos, o puede ser muy paciente y esperar por una oportunidad. Un atacante paciente se encuentra en mejores condiciones de evitar ser detectado diseminando sondeos a todo lo largo del sistema durante un período de tiempo extenso e iniciar dichos sondeos desde diferentes localizaciones.

- **Herramientas** Muchos ataques son llevados a cabo mediante un conjunto de herramientas, lo cual significa que los atacantes pueden tener éxito con un nivel de destreza menor que el que alguna vez se requiriera. Los ataques generados por medio de estas herramientas también pueden tener una única serie de pasos (es decir, una firma que puede ser reconocida por un sistema de detección de intrusión). El atacante sofisticado puede ajustar estas herramientas a fin de modificar la firma y así evitar la detección, o puede desarrollar herramientas con un objetivo específico.
- **Riesgo** Los riesgos que los atacantes aceptan llegar a sufrir a menudo dependen de sus objetivos. Un activista deseará atacar para ser conocido públicamente, y los terroristas pueden reclamar la autoría de un ataque. Por otro lado, un atacante que está tratando de obtener información industrial sensible puede no querer que el objetivo sepa que a tenido lugar un ataque.

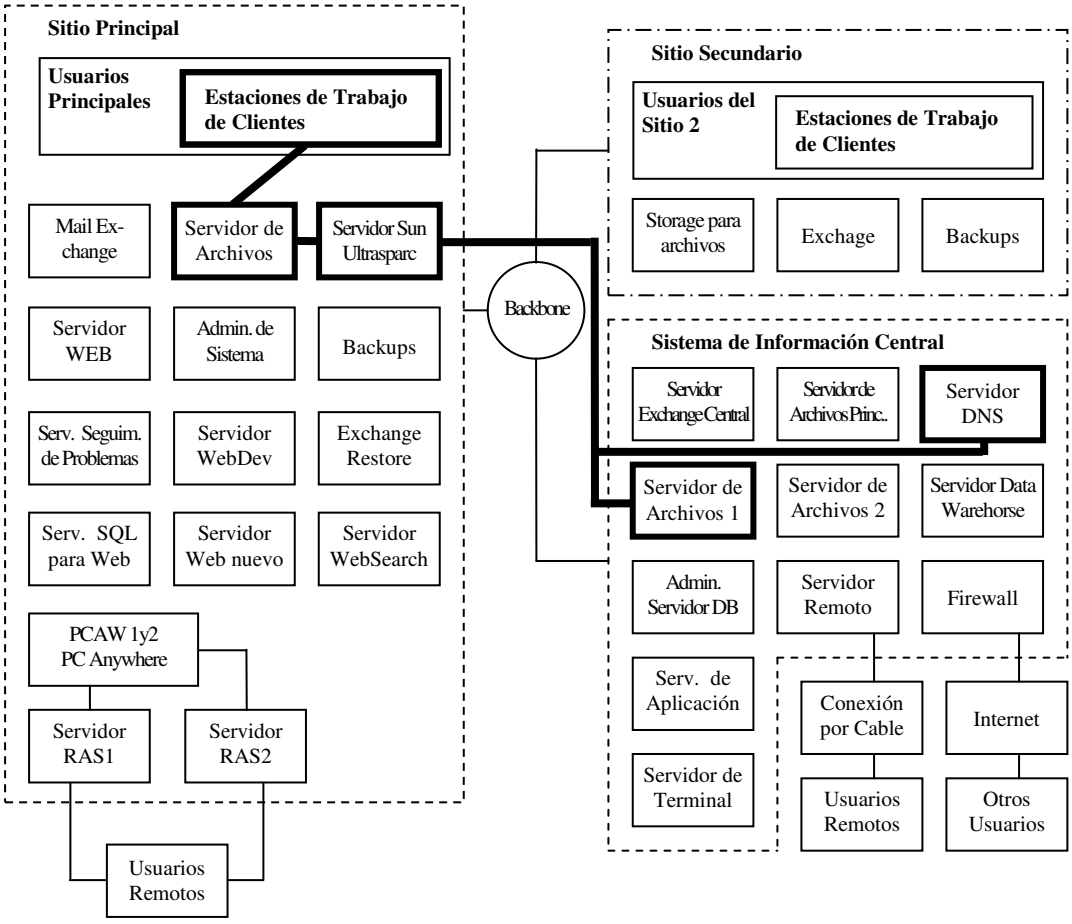


Figura 69. Arquitectura en la que se destacan los Componentes de Servicio Esenciales.

- **Acceso** El acceso de un intruso se describe en términos de mecanismos de acceso, tales como un acceso discado a Internet, y en términos de los límites del sistema, tal como desde dentro o fuera de un *firewall* o una LAN.

- **Objetivos** Los objetivos de un atacante incluyen una ganancia personal, tal como reconocimiento o mejora de sus habilidades de *hacking*, humillar a la organización objetivo, y ganancia económica.

B. Niveles del ataque

Se han considerado tres niveles de ataque: ataques de oportunidad-del-objetivo, intermediario, y sofisticado.

- **Ataque de oportunidad-del-objetivo** El tipo más frecuente de ataque es el denominado "de oportunidad-del-objetivo" y está generalmente asociado con un atacante-por-diversión. Empleados disconformes con algún grado de conocimiento organizacional y habilidades con el sistema también pueden entrar en esta categoría.

  A este tipo de ataques se aplica lo siguiente:

  - El atacante posee un objetivo muy general y, en consecuencia, un amplio rango de objetivos. El impacto inmediato podría ser una denegación de servicio, pero también se podrían ver afectados datos de aplicaciones Web.
  - El atacante utiliza herramientas fáciles de obtener para realizar sondeos y pruebas sobre los sistemas a fin de aprovechar vulnerabilidades conocidas.
  - El atacante puede no requerir acceso de usuario a los sistemas internos –por ejemplo, un ataque podría ser lanzado utilizando un virus de correo electrónico.
  - El atacante posee un conocimiento limitado de los sistemas, procesos y equipo técnicos internos.
  - Existe una alta frecuencia de este tipo de ataques.
  - Los equipos internos involucrados pueden ser utilizados para atacar otros sitios.
  - Estos ataques poseen un impacto relativamente bajo sobre los sistemas bien administrados, pero existen excepciones.

  Las defensas tradicionales contra estos ataques son los *firewalls* para control de acceso, monitoreo de sistemas y aplicaciones para aprender sobre nuevas vulnerabilidades, y actualizaciones del sistema regulares para remover vulnerabilidades conocidas.

- **Ataque intermediario** Los ataques intermediarios generalmente tienen una organización específica como objetivo. Este tipo de atacante realizará los mismos tipos de sondeos y pruebas que un atacante-por-diversión pero puede ocultar mejor esta actividad. Las vulnerabilidades del sistema existentes harán mucho más fácil la tarea del atacante.

  Este tipo de ataque presenta las siguientes características:

  - El ataque puede comprometer inicialmente uno de los sistemas de confianza externos.

- Probablemente el atacante tendrá considerable paciencia y destreza.
- Existe una alta probabilidad de éxito, comparado con un ataque de oportunidad-del-objetivo, y una mayor probabilidad que resulten afectados los servicios esenciales.

- **Ataque sofisticado** El atacante sofisticado tiene un objetivo organizacional muy específico y puede afectar significativamente servicios esenciales. Este tipo de atacante también puede intentar comprometer al equipo técnico interno.

  En tanto la defensa contra un ataque de oportunidad-del-objetivo se podría concentrar en la prevención, los ataques sofisticados, si bien muy poco frecuentes, poseen una alta probabilidad de éxito, y resulta probable que superen medidas preventivas; el aspecto más significativo para una organización es que el objetivo de un ataque sofisticado son la recuperación y el reconocimiento de que el sistema ha sido comprometido.

  Las que se indican a continuación son características de un ataque sofisticado:
  - El atacante será muy paciente.
  - El atacante dedicará un tiempo significativo en recolectar información relativa a la arquitectura y el equipo técnico del sistema.
  - El atacante estará centrado en un objetivo.
  - El atacante puede personalizar o desarrollar herramientas.
  - Existe una alta probabilidad de éxito.

C. Perfiles del atacante

También se consideran en un Reporte del Método SNA los perfiles del atacante que se muestra en la Tabla 17; estos perfiles de atacante pueden ser personalizados para una organización cliente particular.

D. Patrones de ataque

También se consideran patrones de ataque. Los ataques caen dentro de tres patrones generales:

1. **Acceso como usuario**. Los ataques dentro de esta categoría requieren acceso al sistema con privilegios de usuario o de administrador. Los pasos de este tipo de ataque son:
   a. *Reunir información. Realizar una investigación exhaustiva para reunir datos del sistema y para identificar vulnerabilidades de seguridad existentes. Este paso a menudo se encuentra automatizado con herramientas tales como nmap*[43] *o herramientas que apuntan a una aplicación especializada tal como un servidor Web. Las vulnerabilidades podrían existir en los componentes del sistema, pueden ser el resultado de errores en la administración del sistema, o pueden ser el reflejo de políticas de seguridad pobres.*

[43] *nmap* es una herramienta de uso libre que sondea una red. Está disponible en http://www.insecure.com/nmap

Atacante	Recursos	Tiempo	Herramientas	Riesgo	Acceso	Objetivos
Hacker por-diversión	Rangos de destreza Muchos poseen habilidades limitadas Muchos operan como parte de un equipo	Puede ser paciente, pero generalmente buscan la oportunidad	Utiliza conjuntos de herramientas fáciles de obtener	Puede no comprender o apreciar el riesgo	Externo	Reconocimiento personal Desarrollo de destrezas de *hacking*
Empleado disconforme	Depende de destrezas personales Puede poseer conocimiento del proceso Poco probable que utilice recursos externos	Puede ser muy paciente y esperar la oportunidad	Utiliza conjuntos de herramientas fáciles de obtener Antiguos administradores de sistema podrían desarrollar herramientas	Reacio al riesgo particularmente si aún pertenece a la organización	Interno o externo Internet o LAN	Ganancia personal Manchar a la organización
Activista que apunta a una organización por razones éticas o políticas	Medios limitados para contratar expertos externos, pero podría tener miembros talentosos	Probablemente muy paciente, pero eventos específicos pueden forzar a una acción rápida	Utiliza conjuntos de herramientas fáciles de obtener	No reacio al riesgo	Externo Internet	Manchar a la organización Impactar en la opinión pública o de clientes Impactar en el gobierno o en socios corporativos
Espía industrial	Conocimiento de experto	La información deseada tiene un tiempo de vida limitado	Puede personalizar herramientas	Reacio en cierto grado al riesgo Su captura podría impactar en las corporaciones auspiciantes	Externo Internet	Vender información de propiedad exclusiva Obtener conocimiento sobre investigaciones de la competencia, aprender las estrategias corporativas
Seguridad Nacional	Podría contratar recursos externos para un ataque de alto beneficio	Paciente, pero deseada puede necesitársela de manera urgente	Podría desarrollar herramientas si el beneficio es alto	Moderadamente reacio al riesgo, pudiendo operar desde fuera del país	Externo e Internet Podría ser un visitante extranjero	Acceder a información del gobierno o información que es propiedad privada de una organización

Tabla 17. Perfiles de atacante.

b. *Explotar*. Explotar un agujero de seguridad para obtener acceso u obtener información para su uso en un ataque posterior; en las etapas tempranas de un Ataque, la vulnerabilidad puede proveer información tal como nombres de máquinas o nombres de cuentas de usuario.

c. *Daño*. Ocasionar el efecto deseado del ataque a través de medios tales como modificación de datos, acceso a información sensitiva, o establecimiento de una conexión para el acceso continuo. El paso final en este ataque es el intento de cambiar *logs* de tal manera que el ataque no sea identificado.

2. **Acceso a un componente**. Un ataque dentro de esta categoría no requiere del acceso como un usuario al sistema. Estos ataques crean una denegación de servicio mediante el envío de solicitudes inapropiadas. En algunas instancias, tales como una solicitud puede dañar levemente los componentes del sistema deseados. En otros casos, el tiempo extra requerido para procesar una solicitud tal es suficiente como la lentificar de manera apreciable el procesamiento. Los pasos en este tipo de ataque son:

   a. *Reunir información*. Identificar un componente y el puerto de comunicaciones del sistema.

   b. *Explotar*. Envío de mensajes al puerto seleccionado.

   c. *Daño*. Bloqueo o sobrecarga de un componente de aplicación o de un servicio de red.

3. **Contenido de aplicación**. Estos ataques envían datos inapropiados a las aplicaciones en lugar de hacerlo a los componentes de red. En esta situación, el tráfico de red debe poseer un formato apropiado. El problema se presenta con el contenido del tráfico, y al igual que los ataques de acceso a la red, estos ejemplos no requieren que el atacante obtenga acceso de usuario. Los pasos son:

   a. *Reunir información*. Identificar la aplicación objetivo. Esta puede ser una aplicación basada en red tal como un servidor Web o un navegador, o una aplicación tal como *Microsoft Office*, en la que el correo electrónico se emplea para transmitir datos a la aplicación.

   b. *Explotar*. Enviar contenido directa o indirectamente (vía, por ejemplo, correo electrónico) a la aplicación objetivo.

   c. *Daño*. Remover archivos de usuario, modificar la configuración de una estación de trabajo de usuario, y exportar los archivos de usuario. Como ejemplo, consideremos un ataque de virus que sigue el siguiente patrón de contenido de aplicación. El mismo podría ocurrir de la siguiente manera:

      i. *Reunir información*:

- Identificar alias o Ids de usuarios del correo electrónico interno.
- Identificar el software cliente de correo electrónico (por ejemplo, *Outlook*).
- Identificar el navegador Web.
- Conocer cuáles *scanners* se utilizan para detectar virus.
- Conocer la arquitectura de procesamiento de correo electrónico interno en término tanto de servidores como de MTAs –*Mail Transfer Agents*-.

ii. *Explotar*:

- Adjuntan una macro dañina de Visual Basic a un archivo Word o Excel.
- Engañar a un usuario haciendo que descargue un virus desde un sitio Web.
- Utilizar un lenguaje de scripting tal como JavaScript para engañar al usuario haciéndolo ingresar a un sitio, el cual puede capturar el intercambio de información.

iii. *Daño*:

- Remover archivos de usuario.
- Reconfigurar una estación de trabajo de usuario para que soporte un ataque.
- Instalar una "puerta trasera" para comunicaciones en una estación de trabajo de usuario.
- Capturar contraseñas de usuario vía un mailing automático de los mensajes o documentos *Word*.

Sección 6 Recomendaciones

Las Recomendaciones que se hacen en esta sección con respecto a un Reporte de cliente generalmente incluyen un concepto de supervivencia de operaciones, como se muestra en la Figura 70. Las mismas tienden a incluir tanto recomendaciones de política como de arquitectura.

A. Recomendación de política

El siguiente es un ejemplo de una recomendación de política:

> ***Clarificar la política respecto de contenido activo.***
>
> Se ha visto un incremento en los ejemplos de contenido malicioso en los últimos años. Los componentes de políticas podrían variar entre el requerir un filtrado de todo contenido activo que ingrese a una organización hasta la categórica prohibición de ciertas formas de contenido activo. La creación y diseminación de contenido activo, tales como contenido que utiliza Extensible Markup Language (XML), controles Ac-

tiveX, Java applets, y JavaScript por parte de servicios Web tanto internos como externos de la organización, se debería considerar y controlar cuidadosamente.

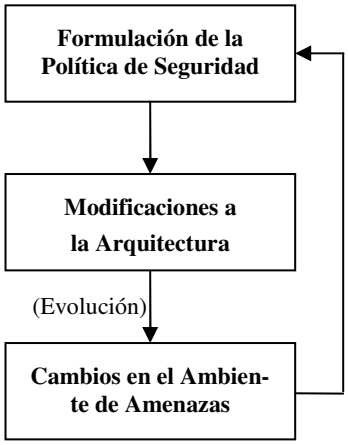


Figura 70. Relación entre Política, Arquitectura y Amenaza.

B. Recomendación de arquitectura

Una recomendación de arquitectura específica para la gestión de virus debería tener la siguiente forma:

Tipo: Aplicación, Red, Infraestructura (sistemas).

Estrategia: Reconocimiento, Resistencia, Recuperación.

Intrusiones: Identificar a cuáles de los escenarios de intrusiones identificados para el cliente se aplica la recomendación (en el ejemplo, los escenarios están identificados por el número).

Fundamento: Provee la justificación y el contexto para la recomendación.

Resultado: Hace una recomendación explícita que puede ser hecha por el cliente.

Implementación: Discute las opciones generales de implementación, riesgos y balance con otras recomendaciones.

El que sigue es un ejemplo de una Recomendación de Arquitectura:

Examinar el contenido Web en los correos electrónicos entrantes y establecer prácticas de gestión de virus.

Tipo: Aplicación

Estrategia: Reconocimiento y Recuperación

Intrusiones: 3, 4 y 5

Fundamento: la inspección de ataques ha identificado que los virus distribuidos vía correo electrónico poseen un muy alto impacto financiero y una muy alta frecuencia de ocurrencia. Los correos electrónicos y las páginas Web cada vez más contienen lo que se denomina “contenido activo”. En el caso del correo electrónico, pueden ser un

documento Word o Excel adjunto que incluye macros. Las páginas Web pueden contener código Java o JavaScript. Los virus tales como *Melissa* de adjuntos que contienen macros dañinas. Un ataque de virus es el medio más común de un ataque desde Internet. Su organización podría recibir un mensaje de este tipo en la fase inicial de un ataque antes que el software de *scanning* de la organización haya sido actualizado para reflejar el incidente más reciente. Un nivel de ataque sofisticado o intermedio crearía un virus que tiene en la mira una organización específica.

Resultado: Establecer prácticas para la gestión de crisis con respecto a virus difundidos vía correo electrónico. Tener pleno conocimiento de los riesgos generales y de las limitaciones de las técnicas de *scanning*.

Implementaciones: Las implementaciones destinadas a mejorar el reconocimiento podrían ser implementadas en múltiples niveles:

a. Realizar un *scanning* de virus en el *hub* de correo electrónico.
b. Realizar un *scanning* de virus en casa servidor Microsoft Exchange.
c. Realizar un *scanning* en el sistema de archivo o computadoras personales portátiles (incluye virus que podrían ser transportados por el acceso Web).

Estas deberían ser las configuraciones de usuario recomendadas para los clientes de correo electrónico tales como *Microsoft Outlook* y navegadores Web utilizados para correo, a los fines de restringir el uso de *scripting* en los mensajes de correo. Se debería advertir a la comunidad de usuarios que soportan vulnerabilidades de virus o de contenidos activos (por ejemplo, *ActiveX*) asociadas con páginas Web y mensajes de correo electrónico.

De igual importancia resultan las mejoras relacionadas con la recuperación. Un virus con *Melissa* puede desbordar a un sistema de correo y crear una denegación de servicio que ese componente esencial. Otros virus pueden remover archivos de usuario o cambiar configuraciones de las estaciones de trabajo. El mejoramiento de la recuperación podría incluir los siguientes pasos:

a. Establecer planes para responder a un virus que no es inicialmente identificado por el software de *scanning* de virus.
b. Establecer o revisar los planes para restaurar una porción significativa de archivos de usuario.
c. Establecer o revisar los planes para restaurar un número significativo de estaciones de trabajo.

Tipo	Corto Plazo: 1-6 meses	Plazo Medio: 6-12 meses	Largo Plazo: 18+ meses
Política	P3: Clarificar la política sobre contenido activo P5: Terminar/modificar los derechos de acceso P7: Eliminar los datos de uso interno del sitio de la Web P8: Revisar la seguridad física P11: Establecer procedimientos de capacitación en seguridad P12: Clarificar los roles de seguridad	P1: Clarificar los derechos de acceso P2: Revisar los dominios de autenticación P4: Definirlos mecanismos de control de acceos P6: *Logging* /monitoreo los *logs* de *hosts* para detectar intrusiones P9: Revisar / practicar respaldo y recuperación P10: Auditar los *drives* C de las estaciones de trabajo y de los equipos portátiles	P13: Desarrollo una política de seguridad
Arquitectura	R1: Agregar firewalls para crear una DMZ R3: Establecer servidores separados para datos internos sensibles y datos externos públicos R5: Eliminar el productor PC Anywhere R6: Agregar *timeouts* para el acceso a datos sensible en estaciones de trabajo R9: Examinar contenido Web en los correos entrantes R10: Establecer defensas basadas en *host* en los servidores Web	R2: Establecer políticas de acceso basadas en el status del empleado R4: Utilizar canales encriptados para acceso remoto R8: Monitorear la integridad de los datos Web públicos R13: Separar el ambiente Web de desarrollo del de producción R15: Monitorear el tráfico de red saliente	R7: Proteger todos los datos de los equipos portátiles R11: Notificar sobre cada actualización de datos sensibles
Operaciones	Procedimiento operacional de implementación de las recomendaciones de política y arquitectura	Procedimiento operacional de implementación de las recomendaciones de política y arquitectura R12: Desviar ataque en la medida de lo posible R14: Establecer un procedimiento para mover datos Web del servidor de desarrollo al servidor de producción	Procedimiento operacional de implementación de las recomendaciones de política y arquitectura

Tabla 18. Línea de tiempo para la planificación de las recomendaciones.

Recomendación	**Labor**	**Equipamiento**
P1: Clarificar los permisos de acceso	Baja	Existente
P2: Revisar los dominios de autenticación	Baja	Existente
P3: Clarificar la política sobre contenido activo*	Baja	Existente
P4: Definir mecanismos de control de acceso	Baja	Bajo
P5: Terminar / modificar los permisos de acceso*	Baja	Existente
P6: *Logging* / monitoreo de *logs* de *host* por intrusiones	Medio	Existente
P7: Eliminar datos de uso interno del sitio Web*	Bajo	Existente
P8: Revisar la seguridad física	Bajo	Bajo
P9: Revisar / ejercitar respaldo y recuperación	Medio	Existente
P10: Auditar los *drivers* C de estaciones de trabajo y portables	Medio	Existente
P11: Establecer procedimientos de capacitación en seguridad*	Alto	Existente
P12: Clarificar los roles de seguridad	Medio	Existente
P13: Desarrollar una política de seguridad	Alto	Existente
R1: Agregar *firewalls* para crear una DMZ*	Alto	Medio
R2: Establecer políticas de acceso en base al status del empleado	Medio	Medio
R3: Establecer servidores separados para datos sensibles internos y datos públicos externos*	Bajo	Medio
R4: Utilizar canales encriptados para acceso remoto	Medio	Bajo
R5: Eliminar el software PC Anywhere*	Bajo	Existente
R6: Agregar *timeouts* en las estaciones de trabajo para el acceso a datos sensibles*	Bajo	Existente
R7: Proteger todos los datos de las portátiles	Medio	Bajo
R8: Monitorear la integridad de los datos públicos de Web	Medio	Bajo
R9: Examinar el contenido Web en el correo entrante*	Alto	Bajo
R10: Establecer defensa basada en *host* en los servidores Web*	Medio	Bajo
R11: Notificar sobre cada actualización de datos sensibles	Medio	Existente
R12: Desviar ataques en la medida de lo posible	Medio	Existente
R13: Separar el ambiente Web de desarrollo y de producción	Bajo	Bajo
R14: Establecer procedimientos para mover datos Web desde un servidor de desarrollo a un servidor de producción	Bajo	Existente
R15: Monitorear el tráfico de red saliente	Medio	Alto

Tabla 19. Recursos relativos estimados para implementar las recomendaciones.

Las recomendaciones también incluyen el desarrollo de un Mapa de Supervivencia, el cual ya se analizara en este reporte.

Sección 7 Implementación

Esta Sección incluye un Diagrama de Avance y una Estimación gruesa de Recursos. Estos se indican en la Tabla 18 y la Tabla 19.

Apéndices y Referencias

Los Apéndices y las Referencias son opcionales, y variarán de un cliente a otro.

ANEXO 2 – TEMAS RELACIONADOS CON LA SUPERVIVENCIA EN EL CONTEXTO DEL SISTEMA

La habilidad de diseñar y desarrollar un sistema con capacidad de supervivencia depende en gran medida de una completa comprensión del contexto en el cual opera el sistema. La característica más sobresaliente de ese contexto es la misión global del negocio para la que el sistema está diseñado dar soporte. En definitiva, es la misión del negocio la que debe sobrevivir, no un sub-sistema, componente, o tecnología en particular [Lipson 99]. La seguridad computacional tradicional se basa en una visión binaria de ataque y defensa, en la que un ataque o es completamente repelido o el ataque tiene éxito y el sistema es comprometido. En el actual ambiente abierto y altamente distribuido de Internet, resulta imposible una defensa perfecta. En contraste, un sistema con capacidad de supervivencia se degrada elegantemente cuando se encuentra bajo ataque, y continúa proveyendo los servicios esenciales aún cuando uno o más de sus componentes hayan sido comprometidos. La supervivencia depende no sólo de la capacidad de un sistema de resistir un ataque, sino también de su capacidad de reconocer los efectos de un ataque, y su capacidad de recuperación frente a ataques que no pueden ser completamente repelidos. La elicitación de los requerimientos de supervivencia, la que incluye la enumeración y descripción de los servicios esenciales que un sistema debe continuar proveyendo de cara a ataques, es una etapa temprana crítica dentro del proceso de desarrollo.

Los arquitectos e ingenieros de sistema también deben estar en conocimiento de las restricciones contextuales impuestas sobre el diseño por factores tales como las redes y las tecnologías existentes que deben funcionar aceitadamente (o al menos adecuadamente) con el nuevo sistema; los problemas de gestión, incluidas las limitaciones sobre la financiación del proyecto, los recursos y el tiempo de terminación; y, por supuesto, un conjunto finito de productos COTS existentes a partir de los cuales se construye el nuevo sistema con capacidad de supervivencia. El uso de productos COTS altamente disponibles reduce costos, pero su diseño genérico, orientado al mercado masivo, y de bajo costo hace poco probable que tales productos satisfagan los requerimientos de supervivencia específicos de aplicación o sistema particular. Una actividad de supervivencia crucial, ya avanzada el proceso de diseño, es comprender los efectos sobre la supervivencia que plantean los productos seleccionados. La falta de disponibilidad del código fuente, de la lógica de diseño, y de otros artefactos de ingeniería asociados con los productos COTS hace que resulte difícil analizarlo en cuanto a la supervivencia, y de otros atributos de calidad del software.

El ambiente de amenazas y los escenarios de ataques potenciales son aspectos adicionales del contexto del sistema que deben ser considerados en gran detalle. La supervivencia es una cali-

dad sensible al contexto, y un sistema no puede ser analizado en cuanto a su capacidad de supervivencia sin una comprensión profunda de la misión, quién y qué probablemente amenaza esa misión, y los escenarios de circunstancias probables bajo los cuales pueden producirse ataques sobre el sistema (y su misión). Por ejemplo, una implementación particular de un algoritmo de encriptación debería ser suficiente para la protección de las transacciones diarias de un comerciante minorista que utiliza Internet, pero podría resultar absolutamente inadecuado para proteger transacciones inter-bancarias. Los escenarios de ataque que involucran un sistema bancario exigirán el uso de una mucha mayor cantidad de recursos de parte de un potencial atacante respecto de lo que exigirían los escenarios para un ataque de un minorista de bajo perfil.

Otro aspecto clave del contexto del sistema tiene que ver con las estrategias de supervivencia de los sistemas externos de los cuales depende el sistema. Tales sistemas externos incluyen infraestructuras locales y globales. Cuando un sistema objetivo se encuentra fuertemente defendido, una estrategia común de ataque es la de intentar afectar los sistemas que proveen servicios del objetivo bien defendido. Por lo tanto, una estrategia de diseño de supervivencia (y el análisis de la gestión de riesgo que el mismo soporta) debe considerar la supervivencia de los sistemas externos, incluidos aquéllos que pertenecer a *partners*, proveedores, clientes y colaboradores del negocio. Las estrategias de la solución de supervivencia deberían especificar medios alternativos de obtener los servicios externos del sistema requeridos el sistema, tal vez con una calidad de servicio degradada pero todavía aceptable.

Los cambios en el contexto del sistema son inevitables, y un sistema con capacidad de supervivencia debe evolucionar a lo largo del tiempo para absorber estos cambios, o la supervivencia y seguridad del sistema se degradará. Un re-examen periódico del contexto del sistema, incluida la tecnología de implementación subyacente, la misión del negocio, los requerimientos de supervivencia, y las políticas y procesos soportados, es una parte crítica de ciclo de vida de desarrollo de un sistema con capacidad de supervivencia. Atender a los resultados de este re-examen periódico es aún más importante para el desarrollo y sostenimiento de los CBSs con capacidad de supervivencia. Primero, las estrategias de supervivencia están basadas en la noción de un adversario inteligente, por lo que las vulnerabilidades en la tecnología de implementación subyacente están continuamente siendo descubiertas como así también nuevos medios para abusar tales debilidades. La distribución intensa distribución de muchos CBSs los vuelve altamente disponibles para la experimentación por parte de *hackers*, y la vulnerabilidad de un producto COTS descubierta en un contexto de sistema puede generalmente se abusada en muchos otros sistemas que emplean ese producto COTS como un componente. Segundo, la liberación planificada de un componente COTS generalmente no se encuentra bajo el control del equipo de diseño de un CBS. La rápida actualización del *release* más nuevo puede ser una necesidad para

continuar satisfaciendo algún requerimiento funcional o no-funcional de sistema, pero las implicancias en cuanto a la supervivencia de la actualización tienen que ser evaluadas (sin el beneficio de los artefactos de ingeniería de la actualización del componente COTS que facilitarían el discernimiento de las implicancias).

Finalmente, un sistema con capacidad de supervivencia no sólo es dependiente de la tecnología, sino también de la política del negocio y de los procesos humanos que soportan la misión global. Los diseñadores y desarrolladores del sistema deben comprender el contexto de la política en la cual opera el sistema, o la supervivencia del mismo están en riesgo.

ANEXO 3 – EJEMPLO DE APLICACIÓN DE TRIAD

A continuación se presenta un ejemplo de aplicación de TRIAD para desarrollar una estrategia de supervivencia a un *e-business* hipotético, denominado ***eBiz***. Este ejemplo trata de ser más ilustrativo que realista, si bien es de esperar que los sistemas del mundo real puedan ser analizados a un nivel estratégico con sólo un modesto incremento de la complejidad.

TRIAD es muy genérico en su naturaleza, dividiendo el espacio de diseño en tres sectores principales. En las secciones 4.3 *Documentación de la estrategia de supervivencia* y 4.4 *Desarrollo de la estrategia de supervivencia* se describieron con mayor detalle las principales actividades que deben ocurrir y los artefactos que necesitan ser documentados en cada uno de los sectores del modelo. Estas actividades y artefactos podrían ser agrupados dentro de un modelo de trabajo específico de muchas maneras. Los detalles del mejor ensamblaje dependen principalmente del dominio de aplicación y de las destrezas del equipo de desarrollo. La Figura 28 describe el proceso utilizado para desarrollar la estrategia de supervivencia para ***eBiz*** dentro del modelo TRIAD de 3 sectores.

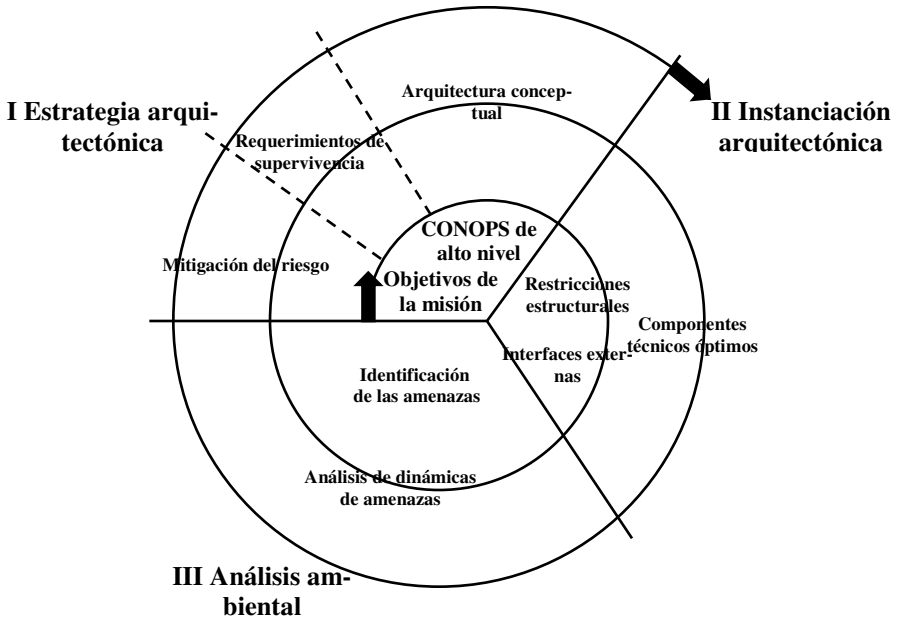


Figura 71. Proceso de desarrollo de la estrategia de supervivencia eBiz.

La Figura 71 muestra dos iteraciones de la espiral TRIAD, culminando en un conjunto final de requerimientos de supervivencia y la arquitectura conceptual que representa la estrategia de supervivencia ***eBiz***. *La primera iteración*, la cual se muestra en el centro de la figura, *comienza con el proceso en un muy alto nivel de abstracción mediante la caracterización de los objetivos globales de la misión, el concepto general de operaciones, y las restricciones estructurales o*

interfaces externas con las cuales cualquier arquitectura debe alinearse. El objetivo de este punto no tiene necesariamente que tener una idea firme de cómo ha de ser asegurada la supervivencia de la misión, sino tan sólo describir qué es lo que necesita el sistema de información para cumplir y alguna idea de cómo esto será llevado a cabo dentro de las restricciones existentes. Las actividades de la Sección III de esta primera iteración comprenden el establecimiento de quiénes se esperan sean los adversarios de la organización y, desde un punto de vista de alto nivel, de qué manera ellos pueden impactar en las operaciones ***eBiz***. Las dinámicas de amenaza resultarán de utilidad en este análisis preliminar.

Luego de determinar de qué manera mitigar las amenazas identificadas, la segunda iteración de la espiral de la Figura 71 se centra en el desarrollo de una arquitectura inicial conceptual. Las tácticas de supervivencia juegan un rol principal en el análisis de mitigación y la derivación de la arquitectura conceptual. Como ya se mencionara, el refinamiento de la estrategia de supervivencia requiere de una cierta cantidad de análisis de factibilidad técnica, para asegurar que la estrategia puede ser implementada de una manera efectiva desde el punto de vista económico. La segunda iteración del Sector II comprende el estudio de las alternativas de los componentes técnicos para asegurar que, de hecho, este es el caso. Luego de posteriores dinámicas de amenazas y análisis de mitigación, se finaliza una arquitectura conceptual basada en el análisis[44].

Justificamos que TRIAD se base en el modelo en espiral debido a que el desarrollo de los sistemas de supervivencia es un dominio en el cual los mejores cursos para el refinamiento son muy poco claros durante las primeras etapas de concepción y refinamiento del sistema. Se requiere de la experimentación y el análisis antes de poder encontrar una solución con un pequeño grado aceptable de riesgo residual de falla de la misión. La especificación y el análisis realizado en cada sector se refinan gradualmente en base a la experiencia de la iteración previa. Por ejemplo, iteraciones subsecuentes de la espiral de la Figura 71, las cuales no se muestran acá, deberían refinar la arquitectura técnica dentro de las restricciones planteadas por la arquitectura conceptual. Existe una posibilidad de que la implementación de la estrategia de supervivencia pudiera ser obstaculizada por detalles técnicos de un nivel menor que no fueron previstos. En este caso, la estrategia de supervivencia puede tener que ser revisada a la luz de esta nueva información. El modelo en espiral soporta específicamente tales revisiones en base a nuevos conocimientos.

El resto de esta sección refina la estrategia de supervivencia ***eBiz*** utilizando el proceso descripto en la Figura 71. Hemos dividido el refinamiento en dos iteraciones de la espiral, seguido por la representación de la arquitectura conceptual final.

[44] La trazabilidad juega un rol importante en el desarrollo, la evaluación y el mantenimiento de la estrategia de supervivencia y su implementación. Sin embargo, no mostramos explícitamente este aspecto del desarrollo de la estrategia de supervivencia eBiz para simplificar su presentación.

A3.1 Primera Iteración

eBiz vende productos a través de Internet. Los productos varían en calidad y costo desde baratos y de baja calidad hasta aquéllos alto costo y primerísima calidad. La mayor parte de las ventas comprenden productos de baja calidad a particulares, pero recientemente se ha producido un incremento en las ventas al por mayor de productos de primera calidad a otros negocios. Todos los pagos a ***eBiz*** se realizan con tarjeta de crédito en línea. Por ello, la misión de ***eBiz*** es mejorar el servicio de ventas de productos de primera calidad a través de Internet de una manera tal que sea tanto lucrativa como legal.

A3.1.1 Operación conceptual

La Figura 72 describe el concepto básico de las operaciones de ***eBiz***. Los clientes compran los productos accediendo al servidor Web de la empresa. ***eBiz*** posee las conexiones usuales a Internet a través de un ISP local. Por razones de seguridad, la Intranet de ***eBiz*** se encuentra protegida de la Internet mediante un *firewall*. Los productos no se pueden enviar a través de la Internet. Un administrador de órdenes de compra asegura que todas ellas han de ser satisfechas

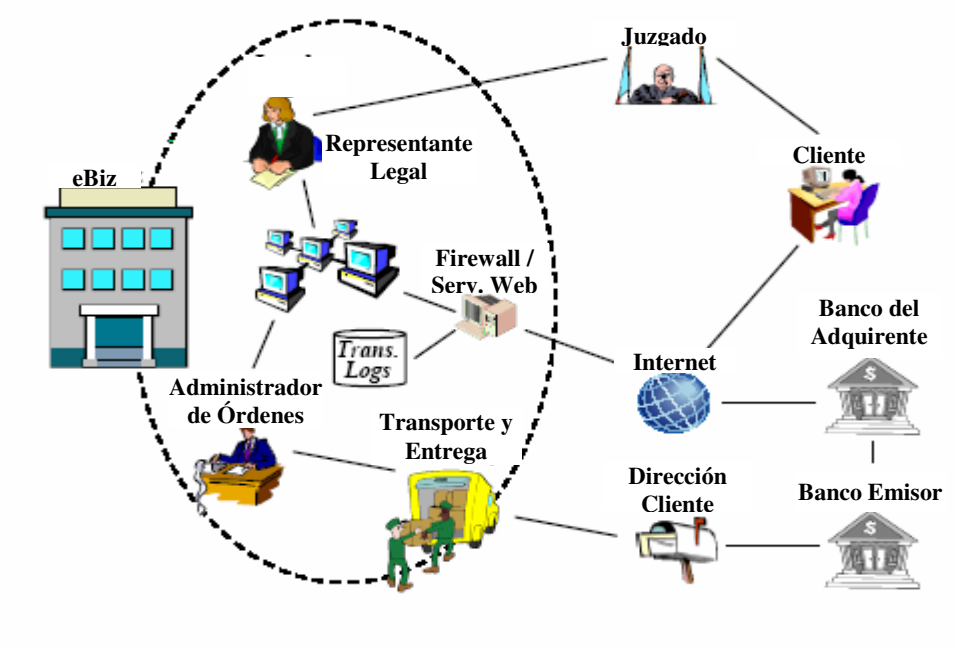


Figura 72. Concepto de operaciones eBiz.

y que un servicio de entrega reparte los productos en la dirección del cliente. En la medida de lo posible, el administrador de órdenes también asegura que todas las transacciones con tarjeta de crédito son atendidas. Desafortunadamente para ***eBiz***, las leyes actuales permiten que la repudiación de compras a través de Internet resulte bastante fácil para el cliente. Se analizarán con mayor detalle las transacciones en línea con tarjeta de crédito y la repudiación que le continúa. Cuando

los clientes tratan negar su acuerdo respecto a la transacción, ***eBiz*** puede presentar una demanda judicial para obtener el pago. ***eBiz*** emplea una representación legal competente y mantiene detallados *logs* de transacciones para ejecutar las demandas judiciales cuando resulta necesario.

El proceso en línea para el pago con tarjeta de crédito es muy similar al proceso tradicional de compra de bienes con una tarjeta de crédito [Hassler 01]. La principal diferencia es que esta transacción se produce a través de Internet. Los clientes deben poseer una tarjeta emitida por una institución acreditada, generalmente conocida como el banco emisor. El banco del comerciante generalmente se denomina el banco adquirente porque es el que recibe los registros de pago, tales como los cupones de compra, de parte del comerciante. Los comerciantes se deben registrar con un proveedor del servicio de pago que conecta la Internet con la red de *clearing* bancario a la cual están enlazadas tanto el banco emisor como el adquirente. Esto constituye la infraestructura que soporta las compras en línea con una tarjeta de crédito, tal como se muestra en la Figura 73.

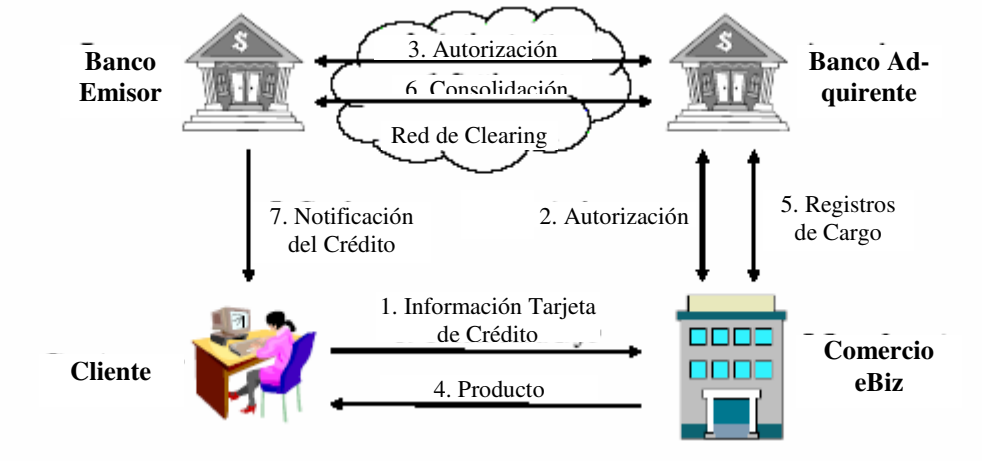


Figura 73. Transacción de pago con tarjeta de crédito en línea.

Hassler describe la transacción típica de pago con tarjeta de crédito de la Figura 73 de la siguiente manera: "El cliente proporciona su información de tarjeta de crédito (es decir, emisor, fecha de expiración, número) al comerciante (1). El comerciante le solicita al banco la autorización (2). El banco adquirente envía un mensaje a través de la red inter-banco solicitándole al banco emisor la autorización (3). El banco emisor envía una respuesta de autorización (3). Si la respuesta es positiva, el banco adquirente notifica al comerciante que ha sido aprobado el cargo. Ahora el comerciante puede enviar los bienes o servicios ordenados al cliente (4) y luego presentar el cargo (o un lote de cargos que representan varias transacciones) al banco adquirente (5 ascendente). El banco adquirente envía una solicitud de aceptación al banco emisor (6 hacia la izquierda). El banco emisor transfiere el dinero a una cuenta inter-bancaria (6 hacia la derecha) y realiza el débito por al cantidad de la compra en la cuenta de la tarjeta de crédito del cliente. A

intervalos regulares (por ejemplo, mensualmente) el banco emisor notifica al cliente sobre las transacciones y sus cargos acumulados (7). Entonces el cliente paga al banco los cargos de alguna manera (por ejemplo, orden de débito directa, transferencia bancaria o cheque). Entre tanto el banco adquirente ha retirado el monto de la venta de la cuenta interbancaria y lo acredita en la cuenta del comerciante (5 descendente)" [Hassler 01].

La Figura 74 muestra la transacción requerida por la repudiación de la compra. Lo que resulta particular es la capacidad del cliente de repudiar los cargos de ***eBiz*** sin mediar ninguna participación de ***eBiz***. Una vez que el cliente niega la autorización de una transacción con tarjeta de crédito (1), el banco emisor automáticamente revoca los cargos sobre la red de clearing, con la completa cooperación del bando adquirente (2 y 4). Éste último notifica a ***eBiz*** de la repudiación (3) y calcula un costo financiero sobre la transacción revocada (6). Una vez que ha tenido lugar el acuerdo final, el banco emisor le notifica al cliente de la revocación del cargo (5).

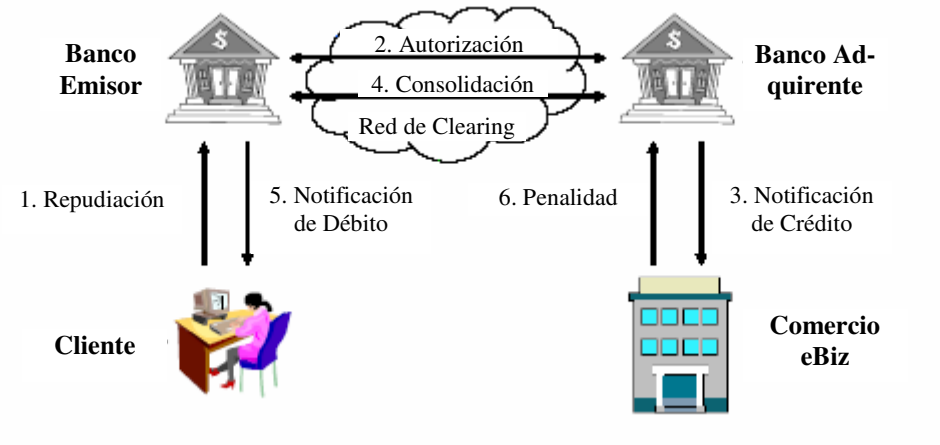


Figura 74. Repudiación de compra.

A3.1.2 Análisis conceptual

eBiz posee un negocio próspero y relativamente consistente, que ha logrado superar la crisis de las ".com". Sin embargo, en el último año ha estado sometido a un creciente número de repudiaciones de compra, es decir, compras de clientes que aparentaban ser legítimas los cuales, más tarde, negaban haberlas realizado. Para ***eBiz***, la detección de compras repudiadas es automática a través de la notificación del banco, pero esto ¿necesariamente indica una actividad fraudulenta? A los fines del análisis, se asume que todas las posibles razones para la repudiación, como por ejemplo, una asignación errónea de cuenta o una entrega equivocada, han sido descartadas o se las considera extremadamente improbables. Las dos razones dominantes para la repudiación de compra son o que se trata de un criminal que ha utilizado información de una tarjeta de crédito robada, o que es un propietario legítimo de la tarjeta, pero deshonesto u olvidadizo, que está tratando de evitar el pago del producto recibido. Estos son los dos principales escenarios que se

han de considerar en el análisis.

Debido a que los principales emisores de tarjetas de crédito responsabilizan totalmente a los *e-business* por los efectos de las repudiaciones de compra, esta tendencia está teniendo un severo efecto en los resultados económicos de ***eBiz***. Raramente se emplean acciones legales cuando los criminales emplean números de tarjetas de crédito robadas para realizar compras en línea, debido a que el rastreo de estos criminales puede resultar extremadamente dificultoso. ***EBiz*** sospecha que la demanda de productos en el mercado en negro los vuelve especialmente atractivos de ser objeto de este tipo de compras fraudulentas. ***eBiz*** necesita reducir drásticamente el volumen de ventas recusadas para poder sobrevivir en un mercado cada vez más competitivo.

Los crecientes niveles de responsabilización que los clientes deben aceptar cuando realizan compras con ***eBiz*** ayudarán a desalentar la actividad fraudulenta, ya se trate de criminales que utilizan números de tarjetas de crédito robadas o de titulares que actúan deshonestamente. El nivel de responsabilización se define como "la propiedad que habilita que las actividades sobre un sistema se encuentren relacionadas con individuos de tal manera que éstos resulten responsables de sus acciones" [NCSC 88]. El nivel de responsabilización por las acciones se ve fuertemente incrementado mediante el empleo de técnicas fuertes de identificación y autenticación, las cuales, en el caso de ***eBiz***, ayudan a verificar que la persona que se encuentra realizando una transacción particular es aquélla que está autorizada a utilizar la tarjeta de crédito. Esta verificación debe permitir su posterior empleo, en caso de ser necesario, por lo que la información de nivel de responsabilización requiere del mantenimiento de algún tipo de *logging* persistente de la transacción.

El nivel de responsabilización se puede utilizar para controlar la actividad fraudulenta que está teniendo en el caso, por ejemplo, que la autenticación y el *logging* actúen como un control de retroalimentación sobre el servicio de ***eBiz***. La Figura 75 describe un diagrama de influencia que caracteriza la influencia creciente del nivel de responsabilización sobre las compras fraudulentas[45]. La figura muestra la tasa de los dos tipo de transacciones fraudulentas con tarjeta de crédito como entrada del elemento "Tasa de solicitudes de compra fraudulenta" en el ángulo inferior izquierdo. Si bien ***eBiz*** no realiza el control de la tasa de robo de información de tarjetas de crédito, esta tasa sin duda que influye en la tasa de criminales que hace uso fraudulento de esa información, como se muestra en la porción superior izquierda de la figura. Un incremento en la tasa de solicitudes de compras fraudulentas tiende a incrementar la tasa de aprobación de compras fraudulentas y, en definitiva, a incrementar la tasa de repudiaciones de compras, como

[45] El texto subrayado que se muestra en éste y los diagramas subsiguientes sirve para proporcionar una explicación adicional que puede resultar de ayuda para comprender o interpretar el diagrama. El texto subrayado puede también describir tendencias de ataques específicas, en aquellos casos en que estas tendencias estén disponibles. Fuerzas externas sobre las que el sistema no tiene control se encuentran encerradas dentro de cajas, como el caso de "Tasa de robo de información de tarjetas de crédito (TdC)", en la Figura 75.

se muestra en el lado derecho de la figura. La flecha de entrada a "Robustez del nivel de responsabilización" indica que ***eBiz*** puede hacer uso de la opción técnicas de nivel de responsabilización como control de retroalimentación sobre la tasa de repudiación. La línea de puntos indica que niveles altos de actividad fraudulenta deberían ser tratados con mecanismos más robustos de de niveles de responsabilización. La línea sólida que apunta a "Tasa de uso criminal de información de tarjetas de crédito robadas" implica que esta acción tenderá a reducir la magnitud de actividad fraudulenta.

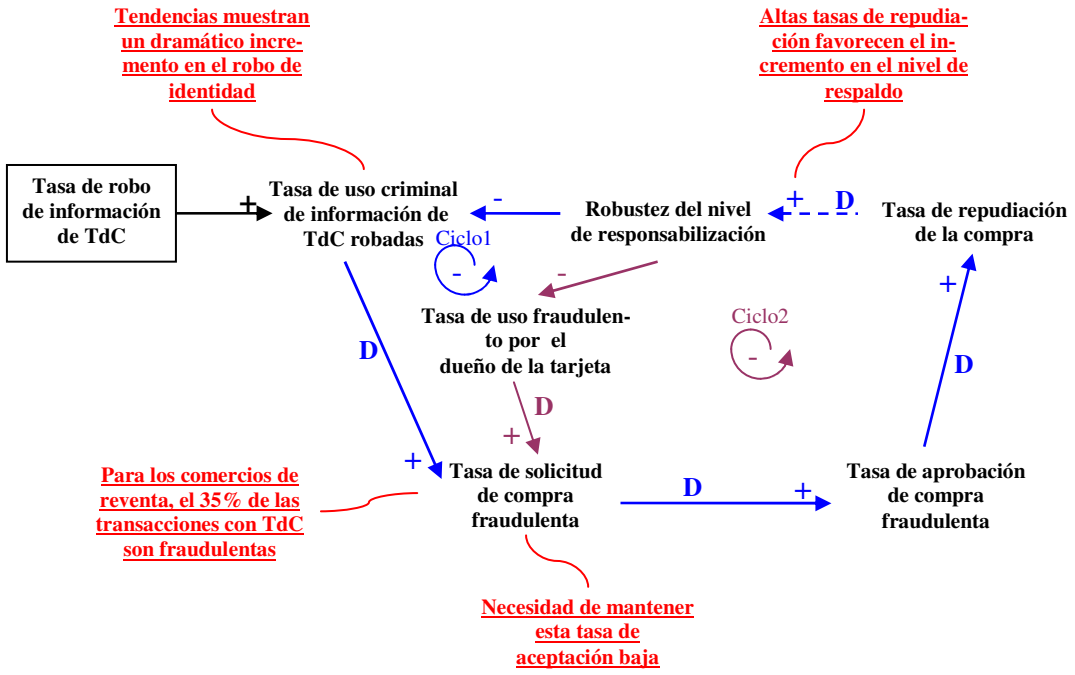


Figura 75. Dinámicas del uso fraudulento de tarjetas.

Volviendo un paso atrás, la Figura 75 muestra dos ciclos de retroalimentación auto-limitantes. El más alejado, etiquetado como Ciclo1, muestra de qué manera se utiliza el control de retroalimentación para limitar la tasa de uso criminal de información de tarjetas de crédito robadas. El ciclo etiquetado como Ciclo2 posee en gran medida la misma estructura que el Ciclo1, pero refleja la influencia de los propietarios deshonestos de tarjetas de crédito sobre la tasa de repudiaciones de compras. Ambos ciclos descansan en la robustez del nivel de responsabilización como control de retroalimentación sobre las tasas de repudiación. Lo que no se muestra en el diagrama de influencia, debido a su naturaleza cualitativa, es el hecho que la robustez del nivel de responsabilización tenderá a evitar el uso de información robada de tarjetas de crédito y a que los propietarios deshonestos de tarjetas piensen dos veces sobre la conveniencia de repudiar la compra luego de haberla realizado.

La Figura 76 amplía los aspectos legales de la opción de uso de mecanismos de responsabiliza-

ción. El Ciclo3 de la figura muestra que la robustez del nivel de responsabilización tiende a incrementar la solidez de las demandas legales contra compras repudiadas, lo que favorece su inicio. El incremento en la tasa de estas demandas, a su vez, aumenta las presiones para mantener o mejorar la robustez de los niveles de responsabilización. Esto demuestra claramente que la fortaleza de estos mecanismos influye en el alcance conque las acciones legales pueden luchar con la actividad fraudulenta. De manera aislada, el ciclo auto-reforzado sostiene la idea que fuertes niveles de responsabilización incrementan las chances de recuperar pérdidas debidas a reclamos repudiados. Desde ya que las acciones legales contra sospechas de fraudes también dependerá del monto de la compra repudiada, dado que las dichas acciones también tienen sus costos asociados.

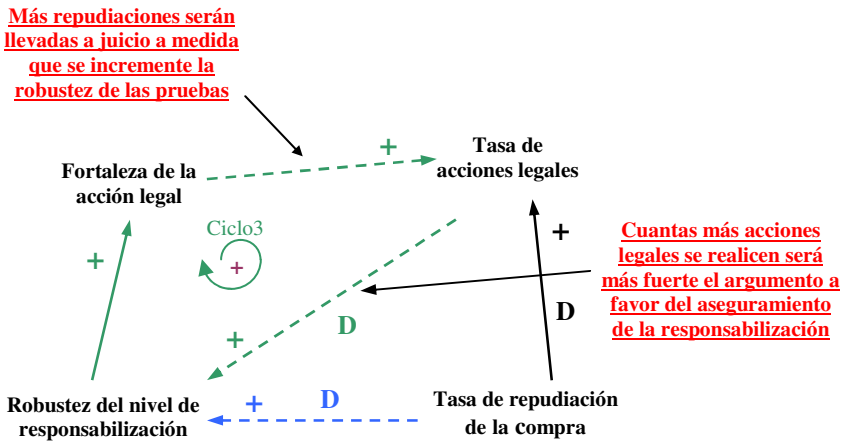


Figura 76. Extensión de las acciones legales.

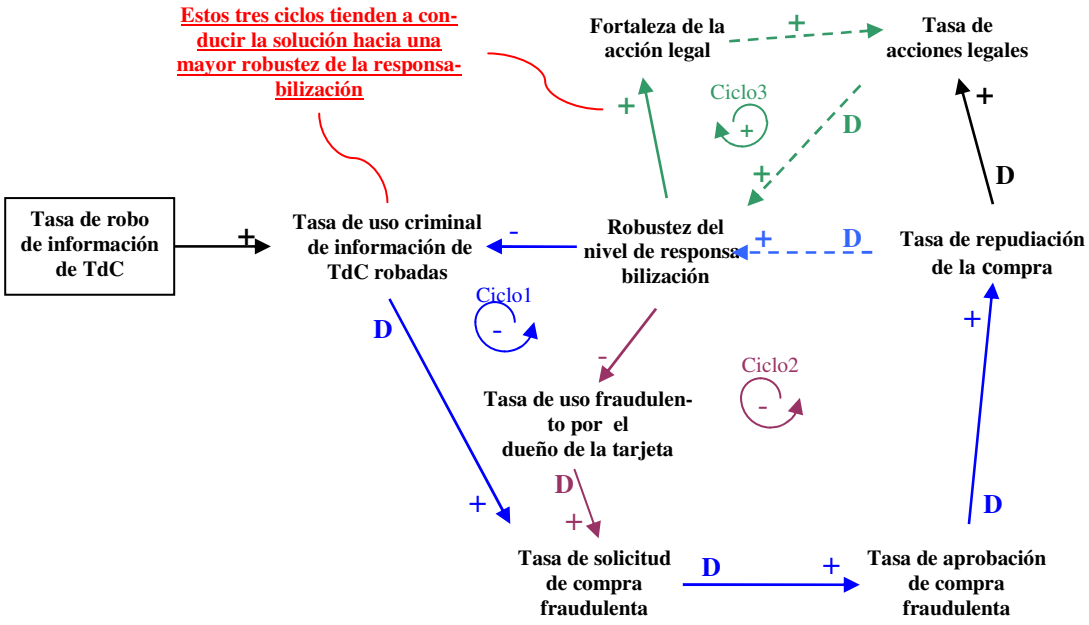


Figura 77. Composición de los Diagramas de Influencia de la primera iteración.

La Figura 77 combina los diagramas de influencia de la Figura 75 y Figura 76. El diagrama muestra que ***eBiz*** puede responder a altas tasas de repudiación mediante (1) incrementando la robustez del nivel de responsabilización para prevenir el uso fraudulento de las tarjetas de crédito y (2) recuperando costos a través de procesos judiciales para los que se cuenta con suficiente evidencias de la existencia de conductas delictivas y el monto de la compra denegada es lo suficientemente alto. La próxima sección describe la segunda iteración del modelo TRIAD, el cual incorpora mecanismos de responsabilización más robustos dentro de la arquitectura conceptual de ***eBiz***.

A3.2 Segunda Iteración

La última sección describió una dinámica de amenazas y un análisis de mitigación que habla a favor del incremento del nivel de responsabilización en las transacciones ***eBiz***. Esta sección describe la segunda iteración de TRIAD, la cual investiga el uso de firmas digitales para el mejoramiento del nivel de responsabilización por parte de los clientes de ***eBiz*** en la misión general del negocio. Las firmas digitales, combinadas con la infraestructura requerida para su uso, constituyen la táctica inicial de la capacidad de supervivencia para enfrentar las compras fraudulentas.

A3.2.1 Arquitectura conceptual refinada

La Tabla 20 presenta los requerimientos de supervivencia iniciales de ***eBiz***. La Figura 78 describe una extensión de la arquitectura conceptual de ***eBiz*** que incorpora el uso de firmas digitales para todas las compras. El propósito de utilizar firmas digitales está orientado a garantizar la no-repudiación de las compras que ***eBiz*** necesita limitar. Los clientes deben firmar digitalmente su solicitud de compra como prueba de que ellos han realizado la orden por el producto. Los bancos deben firmar digitalmente su autorización de pago como prueba para ***eBiz*** que la transacción ha sido aprobada. Desde ya que los clientes han de exigir una prueba del pago, En consecuencia, ***eBiz*** debe firmar digitalmente el recibo de pago que se envíe al cliente. De la misma manera, los bancos pueden exigir una prueba de que ***eBiz*** ha solicitado que el monto de la venta sea acreditado a la cuenta de ***eBiz***. Esto requiere que ***eBiz*** firme digitalmente cualquier solicitud del pago de tarjeta de crédito a la cuenta corporativa de ***eBiz***.

La estrategia descripta anteriormente permite prevenir compras fraudulentas a través de rigurosos mecanismos de responsabilización impuestos por las firmas digitales. La estrategia permite la recuperación de un cierto número de compras fraudulentas a través de demandas civiles. La combinación de firmas digitales certificadas por una autoridad certificante es una fuerte evidencia para este tipo de demandas. Para ***eBiz***, la detección de compras repudiadas (y posiblemente fraudulentas) es automática a través de la notificación del banco. ***EBiz*** debería realizar el seguimiento de las tarjetas perdidas o robadas a través de las "listas de tarjetas calientes", las que en forma habitual distribuyen los bancos. Desde ya que se requiere del mantenimiento cuidado-

so de los *logs* en caso que se deba luego llegar a una demanda legal para recuperar las pérdidas debidas a la repudiación. La prueba de la entrega de la compra en la dirección del dueño de la tarjeta también debería ser necesaria en caso que la compra sea alguna vez repudiada.

<table>
<tr><th colspan="2" rowspan="2">Estímulo</th><th colspan="5">Respuesta</th></tr>
<tr><th colspan="2">Resistencia</th><th>Reconocimiento</th><th colspan="2">Recuperación</th><th>Adaptación</th></tr>
<tr><td rowspan="2">Solicitud de compra en eBiz con intenciones fraudulentas</td><td rowspan="2">Utiliza información de tarjeta de crédito robada</td><td>La compra requiere firma digital</td><td>Bloquear tarjetas que se sabe han sido robadas o perdidas</td><td rowspan="2">Automáticamente a través de la notificación del banco de la repudiación de la compra</td><td>Demanda civil al dueño de la firma digital</td><td rowspan="2">Prueba de la entrega</td><td rowspan="2">Actualizar regularmente la lista de tarjetas bloqueadas</td></tr>
<tr><td colspan="2">La compra requiere firma digital</td><td>Demanda civil al dueño de la tarjeta</td></tr>
<tr><td></td><td>Utiliza su propia tarjeta de crédito</td><td colspan="2"></td><td></td><td></td><td></td><td>No disponible</td></tr>
</table>

Tabla 20. Requerimientos de supervivencia iniciales de ***eBiz***.

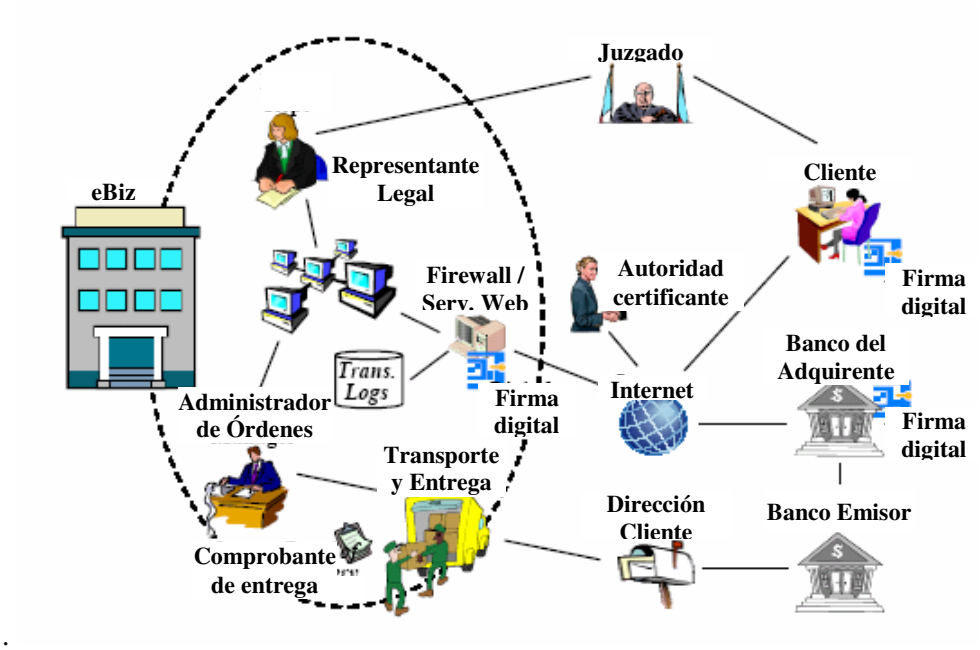


Figura 78. Arquitectura conceptual inicial de eBiz.

A3.2.2 Análisis de la arquitectura conceptual

Si bien los abogados de ***eBiz*** pueden sentirse satisfechos con la anterior estrategia, sus clientes pueden tener una perspectiva distinta. En tanto el fortalecimiento de las contramedidas se puede utilizar como control de retro-alimentación sobre la actividad maliciosa, tal fortalecimiento también puede incrementar los costos asociados. Generalmente, estos costos pueden ser indirectos, tales como el incremento de la complejidad de la administración o la operación del sistema como un todo. En nuestro caso, mecanismos de responsabilización más robustos pueden volver

dificultosa la interacción de los clientes con ***eBiz***, lo que puede desalentarlos! Esto ilustra el necesario equilibrio entre seguridad y usabilidad, el cual se debe alcanzar durante el diseño y mantenimiento del sistema, pero que a menudo sólo se lleva a cabo cuando el sistema se encuentra operativo y los cambios resultan muy costosos de realizar.

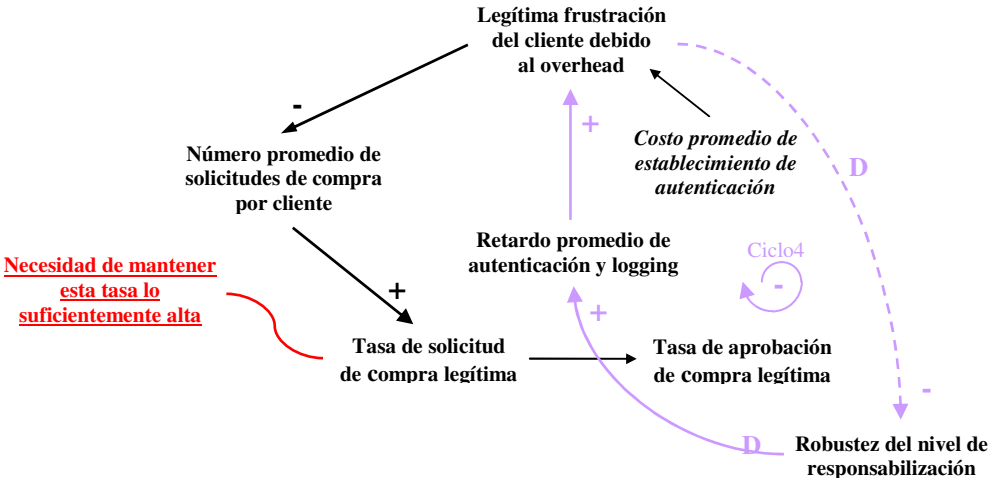


Figura 79. Frustración del cliente debido al fortalecimiento del nivel de responsabilización.

La Figura 79 muestra las principales influencias de la robustez del nivel de responsabilización sobre los clientes legítimos de ***eBiz***. Como se muestra, este fortalecimiento tiende a incrementar las demoras en la aprobación de las compras legítimas, principalmente debido al incremento de las demoras en la autenticación (por ejemplo, verificación telefónica) o la sobrecarga del establecimiento del mecanismo de autenticación (por ejemplo, mediante el uso de terceras partes de confianza). Tales demoras y sobrecarga tienden a incrementar la frustración del cliente frente al proceso de compra, lo cual, a su vez, tiende a resultar en menores compras. Este es un escenario plausible, dado que la arquitectura conceptual inicial de ***eBiz*** requiere de firmas digitales en todas las transacciones para su no-repudiación. La mayoría de los clientes preferirán comprar en otro sitio antes de suscribir con una tercera parte la certificación de firmas digitales que se requieren para realizar negocios con esta compañía. El Ciclo4 de la figura muestra este legítimo nivel de frustración del cliente que se puede utilizar como control de retro-alimentación sobre la fortaleza del nivel de responsabilización. Indicaciones de frustración se pueden evaluar realizando encuestas o monitoreando las tasas de compras, por ejemplo.

La Figura 80 ilustra el diagrama de influencia en el que se integran los dos patrones de respuesta ante amenaza instanciados. También muestran las principales influencia de los elementos del sistema sobre los beneficios de ***eBiz***, como una función del retorno generado y los costos derivados. En resumen, los ciclos etiquetados como Ciclo1, Cicl2 y Ciclo3 describen los aspectos de ***eBiz*** que respaldan un incremento en la fortaleza de los mecanismos de responsabilización tendientes a

alcanzar mayor robustez, mientras que el Ciclo4 describe los aspectos de ***eBiz*** que avalan una menor robustez en estos mecanismos. Debido a que el diagrama de influencia es cualitativo por naturaleza, no describe exactamente qué sucederá a medida que pase el tiempo con la tasa de repudiación de compras, la tasa de compras legítimas, o, más importante aún, su respuesta final. De todas maneras, ayuda a establecer el grado de negociación que debe realizarse para identificar una solución efectiva y abordable al problema de repudiación de compras que padece ***eBiz***.

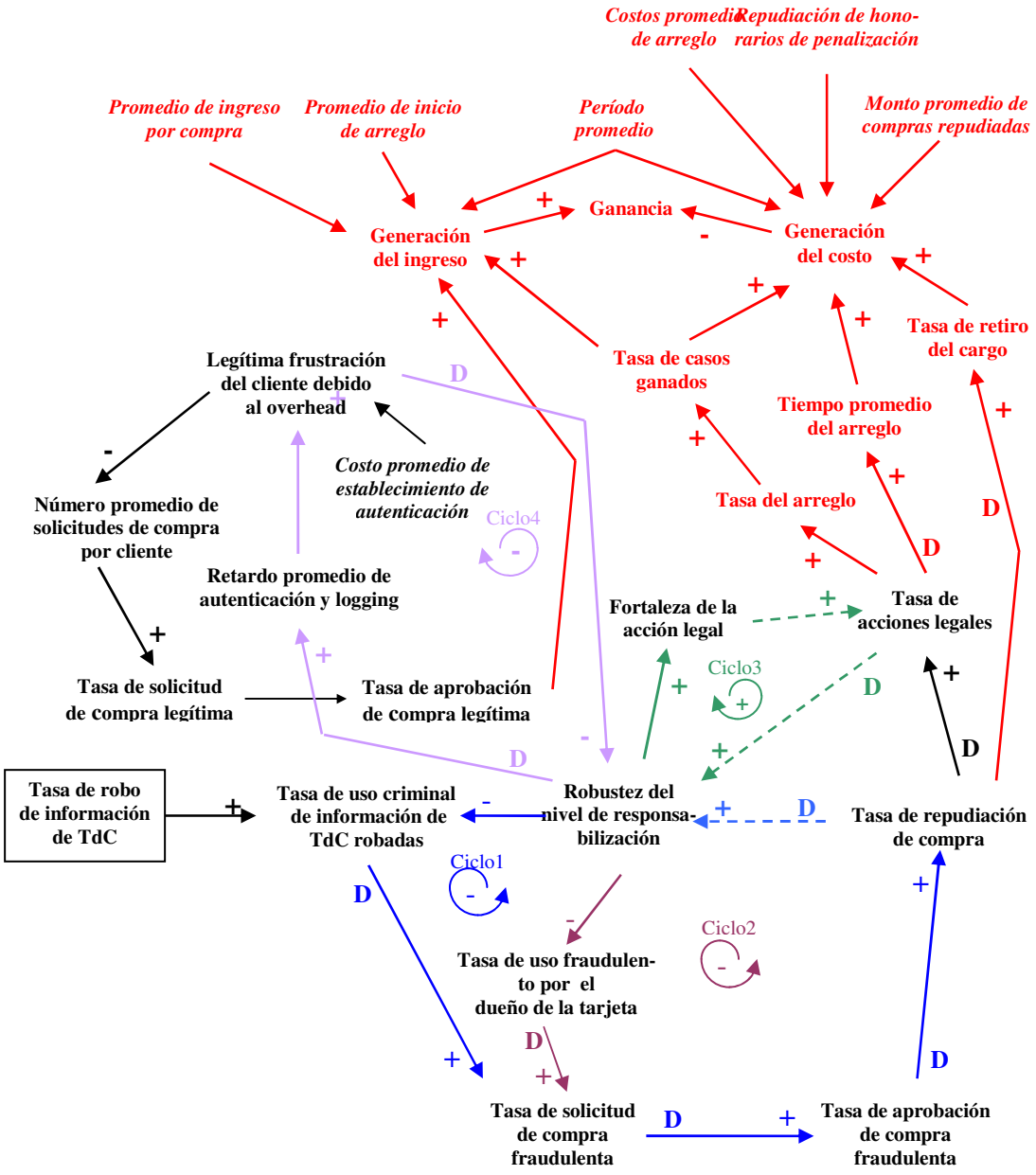


Figura 80. Diagrama de Influencia compuesto.

A3.3 Concepto final

Las dinámicas de amenazas y el análisis de mitigación demuestran que ***eBiz*** no puede afrontar el compromiso de un mecanismo de responsabilización demasiado engorroso. Sin embargo, podría

optar por utilizar mecanismos de responsabilización más robustos para aquéllas compras que superen un determinado monto. Por ejemplo, ***eBiz*** podría adoptar la política de que la solicitud de compra por montos superiores de X dólares deberán ser firmadas digitalmente, o deberá existir una confirmación redundante de la solicitud. ***eBiz*** podría tener que experimentar para establecer los parámetros óptimos de la política. Si X es demasiado alto, ***eBiz*** todavía tendría que seguir soportando un alto nivel de repudiación de compras de menor costo y no podría tener evidencia para una demanda legal que le permita recuperar las pérdidas. Si X es demasiado bajo, la política desalentará las ventas realizadas por personas debido a la alta sobrecarga de la autenticación. Las compras B2B pueden no verse demasiado afectadas por un nivel de responsabilización riguroso, debido a que estos clientes finales probablemente acepten más sobrecarga, tal como certificación de tercera parte de las firmas digitales, para su propia protección.

También se deberían explorar opciones de políticas relativas a la confirmación redundante. Un ejemplo podría ser la verificación telefónica de la solicitud de compra utilizando el número de teléfono asociado con la tarjeta, el cual generalmente está disponible en las compañías de tarjetas de crédito. Además de hacer el seguimiento de tarjetas perdidas o robadas, ***eBiz*** debería hacer el seguimiento de los actores de actos de repudiación, no debiendo esperar que los bancos provean esta función. Se requiere del cuidadoso mantenimiento de registros en caso que sea necesario iniciar demandas legales para recuperar las pérdidas. Una última precaución que debería tomar ***eBiz*** es requerir que todas las ventas sean enviadas a la dirección del dueño de la tarjeta, a menos que se hay recibido una firma digital o se haya realizado una confirmación redundante.

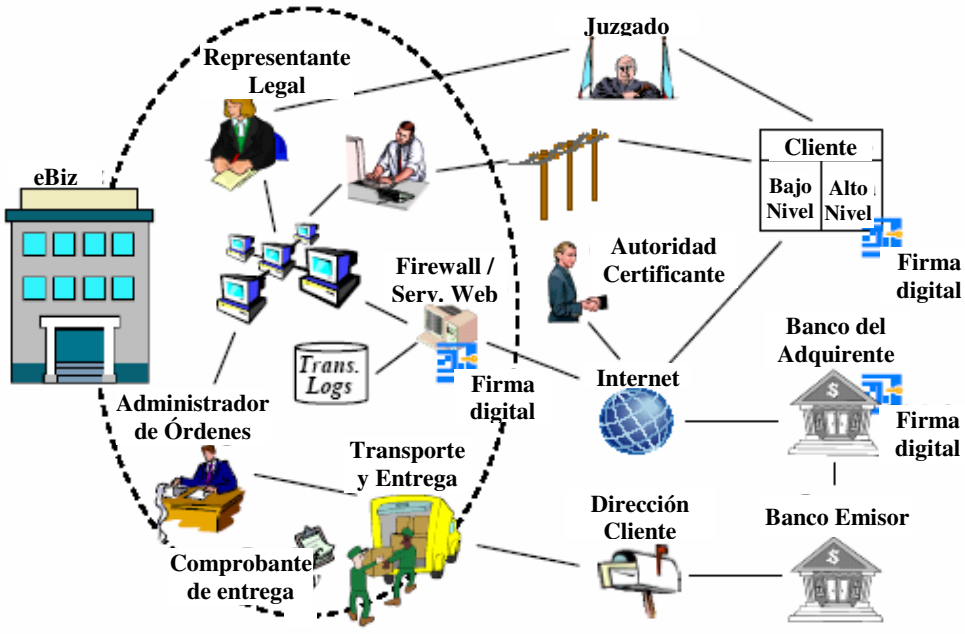


Figura 81. Arquitectura Conceptual final de eBiz.

<table>
<tr><th colspan="3" rowspan="2">Estímulo</th><th colspan="8">Respuesta</th></tr>
<tr><th colspan="2">Resistencia</th><th>Reconocimiento</th><th colspan="2">Recuperación</th><th colspan="2">Adaptación</th></tr>
<tr><td rowspan="4">Solicitud de compra en eBiz con intenciones fraudulentas</td><td rowspan="2">Utiliza información de tarjeta de crédito robada</td><td>Compra < U$S X</td><td rowspan="2">Bloquear tarjetas que se sabe han sido robadas o perdidas</td><td>No disponible</td><td rowspan="4">Automáticamente a través de la notificación de repudiación de la compra</td><td>Enviar sólo a la dirección del dueño de la tarjeta</td><td rowspan="4">Prueba de la entrega</td><td rowspan="2">Actualizar regularmente la lista de tarjetas bloqueadas</td><td rowspan="2">Notificar al dueño de la tarjeta sobre su uso fraudulento</td></tr>
<tr><td>Compra > U$S X</td><td>La compra requiere firma digital o confirmación telefónica</td><td>Demanda legal al dueño de la firma digital</td></tr>
<tr><td rowspan="2">Utiliza su propia tarjeta de crédito</td><td>Compra < U$S X</td><td rowspan="2">Bloquear abusadores de repudios previos</td><td>No disponible</td><td>Enviar sólo a la dirección del dueño de la tarjeta</td><td rowspan="2">Documentar patrones de abusos de repudiación</td><td rowspan="2">Notificar a la compañía de la tarjeta sobre su uso fraudulento</td></tr>
<tr><td>Compra > U$S X</td><td>La compra requiere firma digital o confirma-ción telefónica</td><td>Demanda legal al dueño de la tarjeta</td></tr>
</table>

Tabla 21. Requerimientos de supervivencia finales de ***eBiz***.

La Tabla 21 representa los requerimientos de supervivencia finales de ***eBiz***. La Figura 81 ilustra la arquitectura conceptual final. Los requerimientos de supervivencia y la arquitectura conceptual representan la estrategia de supervivencia para ***eBiz***. Como se muestra, a los clientes del extremo superior se le requiere que o firmen digitalmente sus transacciones o aguarden por una confirmación redundante de su transacción vía teléfono. Evidentemente, no todas las decisiones importantes se pueden llevar a cabo, pero ya se cuenta con una estrategia global. La estrategia permite prevenir, en el grado de lo practicable, la realización de compras fraudulenta sin complicar a los clientes de todos los días con engorrosos requerimientos de seguridad. La estrategia permite la recuperación de ciertas compras fraudulentas a través de demandas legales.

Subsecuentes iteraciones de TRIAD podrían refinar la arquitectura técnica de ***eBiz*** dentro de las restricciones establecidas por la estrategia de supervivencia. Tal refinamiento podría requerir repensar de arriba abajo toda la arquitectura para asegurar que la estrategia se está implementando para dar respuesta a la misión global. Como se describiera previamente, la implementación de la estrategia de supervivencia puede verse obstaculizada por detalles técnicos de nivel menor imprevistos. En este caso, la estrategia de supervivencia tendría que se revisada a la luz de esta nueva información. El modelo en espiral soporta de manera particular tales revisiones basadas en nuevas perspectivas.

ANEXO 4 – EJEMPLO DE V-RATE

Si bien el método V-RATE se encuentra en un estadio temprano de desarrollo, resulta crucial comprender desde un principio que el objetivo de su investigación no es el de asignar un número pequeño de valorizaciones comparadas de productos y procesos de los fabricantes con el propósito de su directa comparación. Lo que se espera es que como resultado de la aplicación del método V-RATE se obtenga un perfil de riesgo del fabricante personalizado para una organización específica que se encuentra interactuando con uno o un conjunto de fabricantes con la finalidad de adquirir, desarrollar, operar o mantener un sistema de misión crítica basado en COTS. Cada elemento de la taxonomía V-RATE representa un componente del riesgo adicional que no está presente en sistemas diseñados a medida. Una apreciación ajustada de estos riesgos (que permite establecer una estrategia para reducirlos) es el principal beneficio que se puede alcanzar al asociar la taxonomía V-RATE con un diseño existente, o propuesto, que incluye componentes COTS.

Sea un sistema de e-commerce que se emplea para realizar compras vía Internet como se ejemplifica en la Figura 82, en el que se incorpora un conjunto de productos COTS, tales como un servidor Web, un firewall, y una aplicación de base de datos. La taxonomía V-RATE puede servir como una hoja de ruta para examinar cada uno de estos componentes utilizados en la arquitectura, pudiéndosela emplear en forma conjunta con las modificaciones arquitectónicas, basadas en estrategias de supervivencia a nivel arquitectónico, para el mejoramiento de la capacidad de supervivencia del sistema. Este proceso es inherentemente interactivo y manejado por el riesgo.

Una vez que se han definido los requerimientos de la misión, el equipo de diseño debería examinar la arquitectura del sistema de e-commerce, existente o propuesta, y modificarla para que ésta soporte las estrategias a alto nivel de supervivencia, dentro del contexto de los escenarios que amenazan a la misión del negocio. En base a este examen basado manejado por escenarios de la arquitectura, los diseñadores pueden decidir, por ejemplo, lo siguiente:

- Adicionar un segundo *firewall* (es decir, una DMZ) para una defensa en profundidad contra ciber-ataques, y un servidor Web de respaldo para hacer frente a eventos de accidente o ataque.
- Desplegar bases de datos redundantes (y diversas) para la recuperación frente a pérdida o corrupción de datos.
- Contratar con ISPs redundantes (y diversos) para contar con una conectividad a Internet con mayor capacidad de supervivencia.

Una vez que se han establecido las estrategias de supervivencia, el equipo de diseño debe asegurar que los componentes utilizados para implementar la arquitectura son técnicamente adecuados. La taxonomía V-RATE se puede emplear para reunir evidencia referida al aseguramiento.

El primer paso es comentar una representación de la arquitectura con los nombres de los fabricantes, y otros atributos del fabricante, para identificar las áreas de exposición. Por ejemplo, es bien sabido que los aparentemente independientes proveedores del servicio de telecomunicaciones ofrecen soluciones de conectividad que comparten la misma infraestructura física. Los diseñadores deberán realizar las preguntas correctas a sus proveedores de servicios (o utilizar otros medios) a fin de asegurar la diversidad.

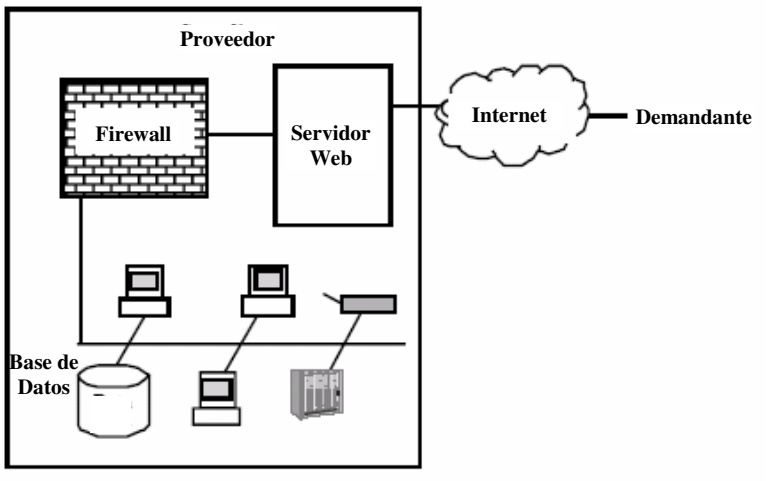


Figura 82. Arquitectura del sistema e-commerce del ejemplo.

A continuación, continuar avanzando con la taxonomía para reunir evidencia acerca del aseguramiento. Primero, considerar el grado de apertura del componente del fabricante. ¿Es posible negociar un acceso total al código fuente y a otros artefactos de ingeniería y diseño (tal vez bajo un acuerdo de confidencialidad)? En tal caso, ¿el equipo técnico posee la experticia suficiente para realizar el análisis y testeo necesario para lograr el aseguramiento requerido? De no se así, probablemente el testeo y análisis de un tercero proporcionará la experticia y el aseguramiento necesarios y podría resultar realmente mejor que darle acceso al propio equipo técnico al código fuente. Este podría ser un ejemplo de una situación en la que el componente COTS puede resultar a uno hecho a medida, en el que el fabricante con alto grado de experticia y de experiencia en un dominio del problema dado desea dejar disponibles los resultados de sus testeos y análisis. Los riesgos relacionados con COTS deberían verse reducidos mediante la verificación detallada de sus respaldos técnicos mediante el testeo y análisis de terceros competentes.

Se continúa de esta manera, paso a paso con la taxonomía V-RATE. En particular, se debe considerar especialmente el valor de los medios no técnicos de mitigación del riesgo, tales como garantías de desempeño, limitaciones de responsabilidad legal, y seguros. El proceso de refinamiento de la arquitectura con supervivencia y de la valoración del riesgo del producto COTS prosigue en forma interactiva en la medida que resulte requerido por las demandas de la misión y las restric-

ciones del negocio. Las valorizaciones numéricas correspondientes a cada categoría (que indican tanto la robustez del aseguramiento y la importancia de esta evidencia para el propietario del sistema) y las valorizaciones combinadas que representan la evidencia acumulada entre categorías, permite contar con valorizaciones relativas de los problemas, con la advertencia de que las valorizaciones combinadas de tales números sólo pueden ser utilizadas con extremo cuidado. Las valorizaciones combinadas adolecen del peligro de que aquellas valorizaciones correspondientes a elementos de la taxonomía importantes para una organización dada sean des-dibujados por otras valorizaciones asociadas a elementos que nada tienen que ver con la misma.

Por otro lado, un perfil de riesgo del vendedor detallado podría proporcionar valorizaciones relativas a un número de elementos de riesgo relevantes para lo que una organización considera importante. Se vislumbran herramientas que provean:

- Vistas o perspectivas múltiples de un perfil de riesgo del vendedor.
- La posibilidad de que un conjunto de elementos de taxonomía agrupados o no produzcan un amplio rango de representaciones variadas.
- Vistas que resulten adecuadas para análisis del tipo “qué si” y comparaciones de alternativas de diseño o adquisición.

Sin embargo, las valorizaciones numéricas específicas y los perfiles de riesgo del fabricante que resulten del proceso V-RATE no son tan importantes como lo que se aprende a lo largo del proceso en sí mismo. Resulta posible reducir el riesgo de manera significativa si el propietario del sistema realiza el ejercicio de asignarle importancia a cada categoría V-RATE y evaluando los riesgos asociados con los productos y procesos del fabricante, como así también sus propios procesos de gestión. Se pueden lograr mayores beneficios si te tiene el respaldo del fabricante durante este ejercicio.

ANEXO 5 – TEMAS RELATIVOS A LA INGENIERIA FSQ

A5.1 Teoremas FSQ

A5.1.1 Teorema de la Estructura de Flujo

Dado cualquier grafo que represente a un flujo, existe un flujo equivalente que puede ser implementado utilizando solamente estructuras de control de composición, alternancia e iteración.

Implicancias en la ingeniería: Este teorema garantiza que las estructuras de control de composición, alternancia e iteración son suficientes para implementar cualquier flujo. En consecuencia, los desarrolladores de flujos no necesitan utilizar lógica no-estructurada ni delegaciones arbitrarias.

A5.1.2 Teorema de la Abstracción / Refinamiento

Dos flujos F y G son equivalentes en el ambiente de cualquier red si y sólo si tienen especificaciones de flujo idénticas.

Implicancias en la ingeniería: Este teorema es la justificación básica de que las semánticas matemáticas FSQ son correctas. El mismo consta de dos partes, necesarias y suficientes. Lo suficiente establece que dos flujos cualesquier que tienen la misma especificación matemática pueden ser intercambiados sin afectar los resultados de cualquier ambiente de red mayor. Lo necesario establece que para dos flujos cualesquiera que poseen especificaciones matemáticas diferentes, existen dos ambientes de red mayores que producirán diferentes resultados si se los intercambia. En esencia, este teorema dice que todo lo que sea importante acerca del comportamiento de un flujo desde el punto de vista de la red externa está contenido en su especificación.

A5.1.3 Teorema de la Verificación del Flujo

El conjunto básico de estructuras de control única-entrada, única-salida que comprende los diseños de flujos se puede verificar mediante la evaluación de las ecuaciones funcionales que se muestran a continuación correspondientes a las estructuras representativas. Cada ecuación está seguida por una sentencia "Pregunta de Corrección" que articula las condiciones de verificación. Los requerimientos de verificación para estructuras de control similares son fácilmente derivables.

A los fines de soportar la función composición de especificaciones de flujos en las operaciones de verificación, sus dominios y rangos se deben extender a un super-conjunto común. En el caso de una especificación de flujo [F] = *f:* IxIRH → OxISH, primero se extiende *f* a la función *f'* = SxIRH → SxISH, donde S es el espacio de estado de verificación funcional correspondiente a los parámetros de entrada y de salida y a las variables del flujo F. Luego, *f'* es extendido a la función *f":* SxIRHxISH → SxIRHxISH, dada por *f"*(*s,rh,sh*) = (*s',rh',sh'*), donde (*s',sh'*) =

f'(*s,rh*), y *rh'* es igual a *rh* con los primeros n elementos removidos, donde *n* es la longitud de *sh'*. Al hacer esta extensión, las semánticas de un flujo completo se pueden calcular mediante una tabla de traceo de verificación funcional con una columna extra para la variable *historia de respuesta rh*. En consecuencia, para la especificación de un flujo *f*, los flujos *g* y *h*, predicado *p*, y para todos los posibles argumentos de F, las estructuras de control se pueden verificar de la siguiente manera [Mills 88, Prowell 99]:

- Estructura composición:

  f = *g*; ¿*g* está seguida por *h do f*?

- Estructura alternancia:

p → *f* = *g* | ~*p* → *f* = *h*;	Siempre que *p* es verdadero, ¿ es *g do f* ? y Siempre que *p* es falso, ¿ es *h do f* ?

- Estructura iteración:

Termination ^	¿Termina la iteración? y
p → *f* = *g* o *f*^	Si *p* es verdadero, ¿ es *g* seguido por *f do f* ? y
~*p* → *f* = *null*	Si *p* es falso, ¿ es no hacer *nada do f* ?

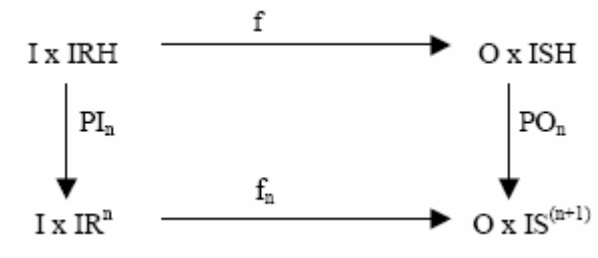
Implicancias en la ingeniería: El teorema de la Verificación de Flujos reduce al verificación a un proceso finito y completo a pesar del hecho que los flujos pueden contener un número virtualmente infinito de caminos. La verificación puede ser llevada a cabo utilizando las Tablas de Traceo [Prowell 99] como un proceso formal de análisis y documentación para las partes críticas del sistema, o ser realizada mediante revisiones de equipo con una mayor velocidad y menor pérdida de precisión.

A5.1.4 Teorema de la Implementación del Flujo

Para cada función computable *f* desde IxIRH → OxISH que satisface la siguiente condición, existe un flujo F tal que [F] = *f*.

Condición: para todo $n > 0$, existe una función f_n: $IxIR^n$ → $OxIS^{(n+1)}$, donde IR^n son los primeros *n* elementos de IRH, y $IS^{(n+1)}$ son los primeros $(n+1)$ elementos de ISH, tal que el siguiente diagrama conmute, donde PI_n: IxIRH → $IxIR^n$ está dado por $PI_n(i,rh)$ = (*i*, primeros *n* elementos de *rh*), y PO_n: OxISH → $OxIS^{(n+1)}$ está dado por $PO_n(\phi,sh) = (\phi,sh)$ si *sh* tiene *n* o menos elementos.

Implicancias en la ingeniería: Cada flujo F posee una función [F], pero resulta que no toda función *f* corresponde a un flujo. Este teorema define cuáles funciones corresponden, de hecho, a los flujos. Informalmente, los estados de condición que una función no puede requerirle a un flujo para predecir el futuro, mediante la realización de decisiones basadas en respuestas obtenidas de los servicios antes que estos servicios fueran realmente invocados. El teorema establece que cualquier semántica de flujo que satisfaga esta condición puede realmente ser implementada. En consecuencia, este teorema es importante para tener en cuenta cuándo el diseño avanzada de manera *top-down*.



A5.1.5 Teorema del Testeo del Sistema

Supongamos que D_I es una distribución de utilización sobre la salida de un flujo F, y que D_R es una distribución de utilización sobre las respuestas a las llamadas de servicio externo realizadas por F. Entonces, la distribución de utilización D_S sobre el estímulo de las llamadas de servicio externos por F se pueden calcular a partir de D_R, D_R y [F].

Implicancias en la ingeniería: Los sistemas, cualquiera sea su tamaño, exhiben una población virtualmente infinita de posibles ejecuciones. En consecuencia, todos los testeos son muestras, y la única pregunta real es de qué manera diseñar la muestra con los casos de testeo que ha de ser ejecutada. Si una muestra es representativa del campo de utilización real, se pueden realizar predicciones válidas científicamente en base al testeo de muestra para la población de todas las ejecuciones que no han sido testeadas, en las cuales, por supuesto, los usuarios han de encontrarse dentro del campo de utilización.

Este moderno testeo estadístico basado en la utilización soporta efectivos procesos de gestión de testeo y reducción del riesgo. Las especificaciones de flujos y los procesos efectivos de testeo del sistema están íntimamente relacionados. Los flujos definen cómo se utilizan los sistemas. Dadas las frecuencias de utilización para los flujos que definen cuándo y cómo se los ha de utilizar, resulta posible predecir la utilización de sus servicios. Tales predicciones pueden poblar las distribuciones de probabilidad de los modelos de utilización que son empleados para generar los casos de testeo (muestras de la población de ejecución) estadísticamente fieles para la utilización anticipada. Tal solución para el testeo permite estimar de manera válida el desempeño del sistema en el campo de uso y, en consecuencia, guía las decisiones de gestión de testeo y de lanzamiento del producto.

Otros teoremas proveen fundamentos adicionales para las operaciones de ingeniería FSQ, incluidos el análisis transitivo de las dependencias de flujo y la derivación de las arquitecturas de sistema lógicas a partir de los flujos.

A5.2 Operaciones de Ingeniería de Estructuras de Flujos

A5.2.1 Abstracción y refinamiento de flujos

El conjunto básico de estructuras de control de única-entrada, única-salida que comprende FSL

puede estar compuesto y anidado para crear Estructuras de Flujos de cualquier tamaño y complejidad. Como se indicara anteriormente, los flujos y sus estructuras de control constitutivas implementan funciones o relaciones matemáticas, esto es, asociaciones desde dominios hacia rangos. Estas funciones definen los datos que se transfieren desde sus valores iniciales hacia sus valores finales, y se los puede incluir dentro de los flujos como comentarios adjuntos a sus refinamientos de las estructuras de control. Se los puede definir como una variedad de formularios, que varían desde lenguaje natural hasta notaciones basadas en matemática.

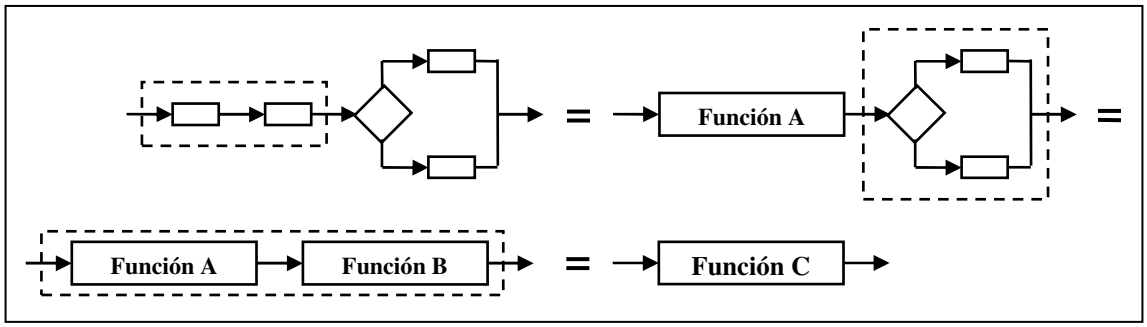


Figura 83. Operaciones algebraicas en el Análisis y Diseño de la Estructura de Flujo.

Un álgebra de funciones permite una abstracción y un refinamiento precisos en la sustitución de las definiciones de función para sus refinamientos (abstracción para determinar cuáles flujos realmente realizar) o sustituciones de diseños para sus definiciones de función (refinamiento para implementar cuáles flujos se presenten hacer). La Figura 83 describe estas operaciones en un formato abstracto. La abstracción del diseño tiene lugar yendo de izquierda a derecha, en la que la función global del flujo está abstraída en tres etapas de la función C. El refinamiento del diseño tiene lugar yendo de derecha a izquierda, en la que la elaboración completa de la función se logra en tres etapas.

La Figura 84 provee una ilustración ficticia en miniatura del refinamiento y abstracción de flujo basado en el ejemplo de la transacción de compra de combustible. Esta descripción informal muestra el flujo a nivel de abstracción de la misión, la que está refinada como una especificación de las operaciones principales y de su composición. La primera operación especificada está refinada más aún en un diseño de alto nivel. La abstracción revierte este proceso que termina en la descripción de la misión del flujo [Hausler 90, Pleszkoch 90].

En dicha figura se describen los métodos informales de abstracción y refinamiento. Sin embargo, las semánticas de *Estructuras de Flujo* permiten que estas operaciones sean realizadas con precisión, para soportar el ensamblado previsible de componentes en las operaciones de composición, y para permitir la verificación de los componentes con respecto a sus especificaciones.

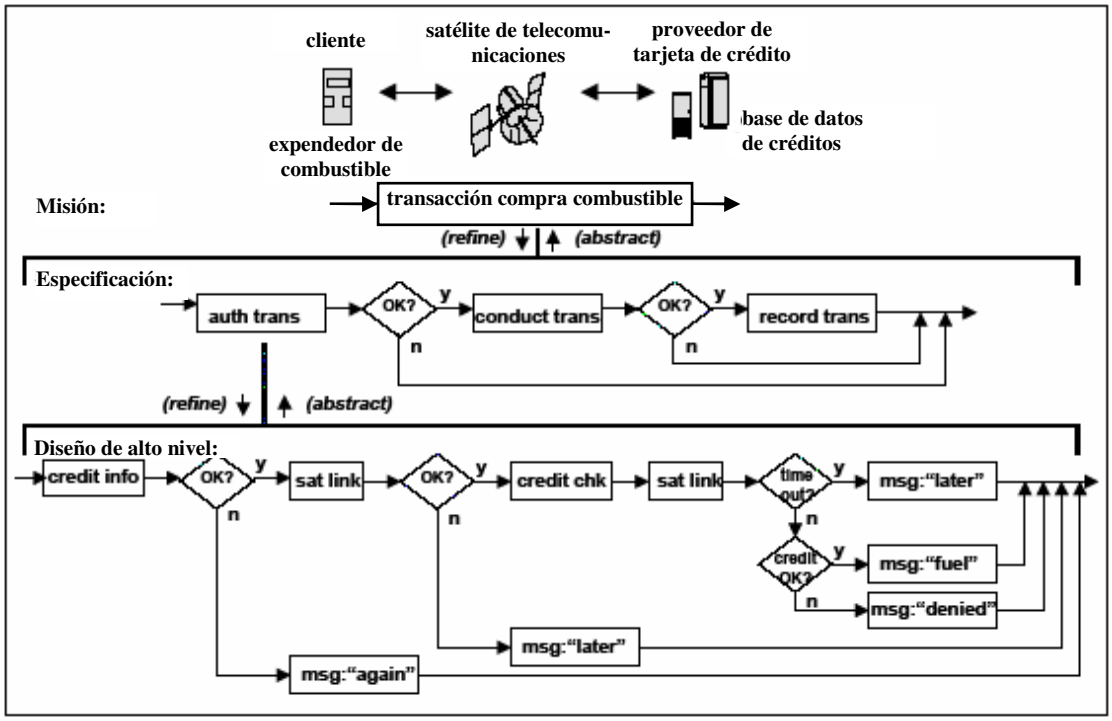


Figura 84: Abstracción y refinamiento de flujo en un sistemas transaccional.

A5.2.2 Verificación de flujos

El teorema de Verificación de Flujos proporciona la base para un efectivo proceso de aseguramiento orientado a equipo. Los flujos comentados con definiciones de función de sus estructuras de control constituyentes pueden ser verificados mediante revisiones estructuradas de equipos que preguntan y responden las "Preguntas de Corrección" y que requieren de un acuerdo unánime sobre cada pregunta.

Los flujos pueden contener un número virtualmente infinito de posibles caminos de ejecución que deben ser verificados en forma individual. No obstante, los mismos están compuestos por un número finito de estructuras de control, y el Teorema de Verificación de Flujos permite que cada estructura de control sea verificada realizando un número finito de etapas [Mills 86, Prowell 99]. Se requiere de una etapa para la verificación de la secuencia (función de composición), dos etapas para la verificación de alternancia (análisis del caso verdadero/falso) y tres etapas par la verificación de la iteración (prueba de terminación, sumado análisis del caso verdadero/falso y de la función de composición). En consecuencia, la verificación se reduce a un proceso finito asequible para las operaciones del equipo.

Estas verificaciones del equipo resultan efectivas desde el punto de vista del costo al permitir la detección temprana de errores y problemas en el desarrollo y dar por resultado una corrección a menor costo. La Figura 85 ilustra el proceso de verificación del flujo. Como se muestra en la

izquierda, se ha de verificar un flujo compuesto por una secuencia seguida de una alternancia. Se debe verificar la corrección de la secuencia, la alternancia y la composición de estas dos estructuras con respecto a las funciones deseadas. En el centro de la figura se indican los errores descubiertos en base a la aplicación de las "Preguntas de Corrección", y se los muestra corregidos en el lado derecho.

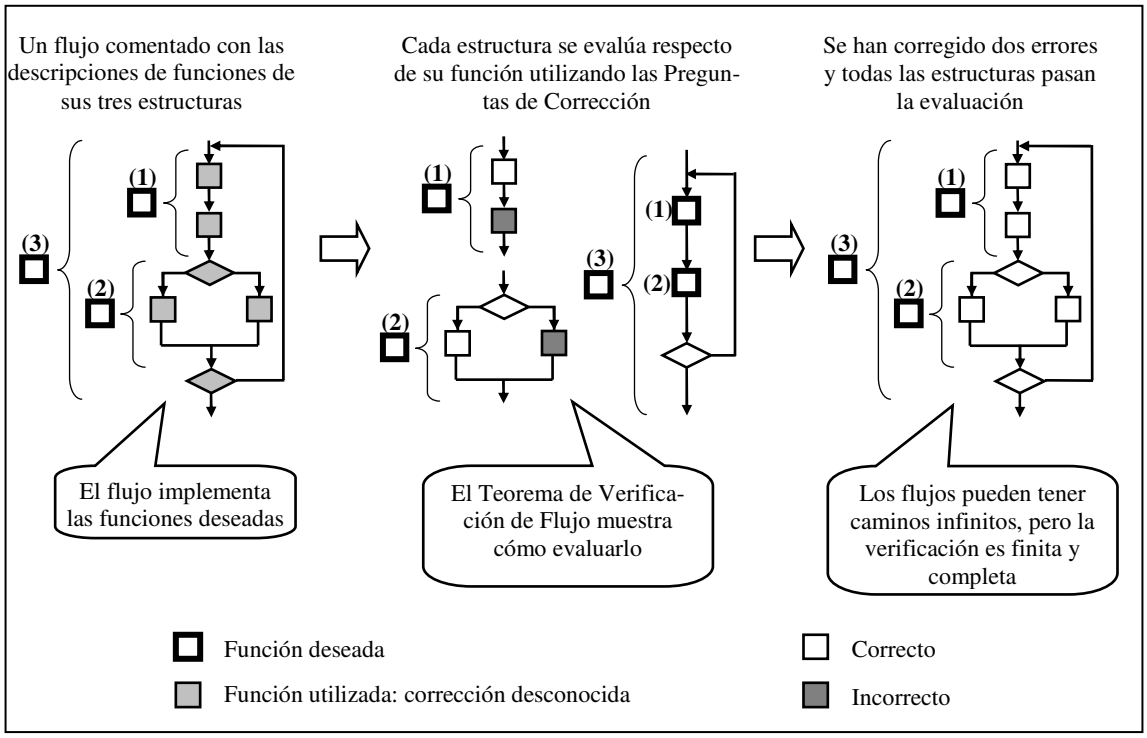


Figura 85. Proceso de evaluación de corrección basado en el Teorema de Verificación de Flujos.

Cada verificación de una estructura de control contra su función deseada sólo requiere un razonamiento local, y las verificaciones se pueden realizar en cualquier orden. Cuando se requiere de mayor precisión para el caso de flujos críticos, la verificación se puede realizar utilizando tablas de traceo para documentar el razonamiento y el análisis. Cuando un servicio se provee mediante un componente COTS con funcionalidad compleja, solamente resulta necesario verificar el uso que realmente se hace del componente. Esta verificación incluirá las evaluaciones de equivalencias de clase que toman en cuenta todos los posibles resultados de la invocación del COTS.

A5.2.3 Análisis de transitividad de flujo

Los flujos pueden invocar servicios constituidos por flujos, los que a su vez también pueden invocar servicios compuestos por flujos a un nivel inferior, etc. En consecuencia, un flujo principal puede depender para su realización de muchos otros flujos, posiblemente distribuidos a través de múltiples componentes dentro de una red de gran escala. El análisis de transitividad de flujos puede revelar tales dependencias para ser consideradas en el análisis de supervivencia y

de otros atributos de calidad.

La Figura 86 describe el comienzo de este tipo de análisis para el ejemplo del Sistema de Combate del Futuro. El flujo de control principal para la misión ubicado en el centro de la figura, denominado Atacar Objetivo invoca a un servicio sensor de datos y su flujo es provisto por un nodo UAV, y a un servicio de control de disparo y su flujo provisto por un nodo de disparo dirigido por robótica. Estos flujos, a su vez, invocan otros servicios locales a sus nodos. Por lo tanto, la primera etapa en el análisis de transitividad consiste en determinar cuáles servicios y sus flujos son directamente invocados para satisfacer un flujo principal.

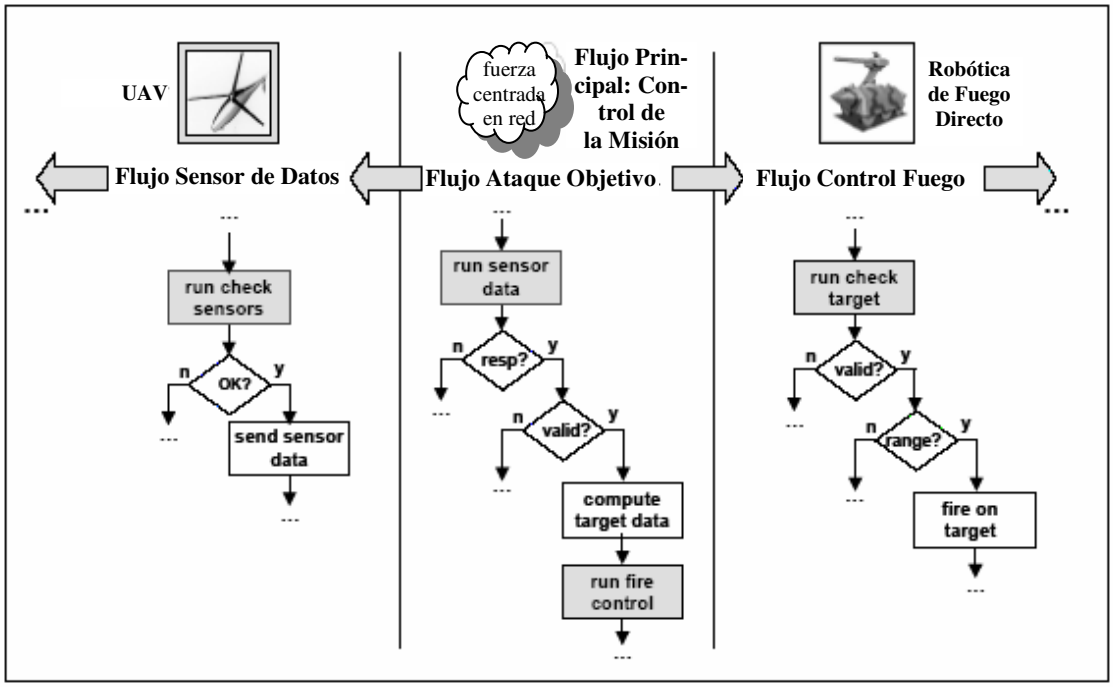


Figura 86. Análisis de transitividad de las dependencias de flujos.

Además, cada flujo puede exhibir tanto resultados deseados como indeseados como está definido por las equivalencias de clases sobre las respuestas de servicios. Evidentemente, lo que se intenta hacer en una invocación de flujo es un resultado deseado. Resulta frecuente el caso que los resultados deseados dependen de la terminación exitosa de flujos no invocados directamente. Por ejemplo, para ser lo más general posible, un componente UAV debe haber experimentado una terminación exitosa de los flujos relacionados con inventario, entrenamiento, mantenimiento, análisis del clima, definición de la misión, abastecimiento de combustible, lanzamiento y vuelo, por nombrar algunos, a los fines de un resultado deseado para poder obtener cuándo invocar el servicio de sensor de datos. Esta cadena de dependencias representa el conjunto completo de flujos UAV que sustentan la misión del flujo principal de Atacar Objetivo.

Este análisis puede revelar sorprendentes y a menudo inesperadas dependencias que deben ser consideradas para asegurar la supervivencia de los flujos de la misión. Los flujos importantes para los objetivos de la organización pueden ser vulnerables a los flujos pobremente diseñados e implementados que de otra manera difícilmente podrían ser percibidos debido su separación física o temporal de los flujos principales.

Los flujos orientados a la misión conforman e integran los servicios de red local dentro de capacidades coherentes que son la razón de la existencia del sistema en red antes de nada. Como tales, ellos sirven como un puente entre las especificaciones a nivel de red y los servicios locales que las soportan.

A5.2.4 Conjuntos de flujos en sistemas a gran escala

Los sistemas a gran escala generalmente contienen componentes que son sistemas complejos en sí mismos. Como tales, estos componentes soportan servicios y flujos que gestionan y utilizan sus propias capacidades, como también participan en flujos inter-componentes que abarcan la red. A menudo resulta de utilidad agrupar las servicios y flujos que soportan componentes o funciones particulares. Tales agrupamientos se denominan **Conjuntos de Flujos** (*FlowSet*).

Por ejemplo, consideremos los Conjuntos de Flujos asociados con el Sistema de Combate del Futuro como se describe en la Figura 87. Cada nodo sensor y de entrega de armas es un sistema de gran escala que exhibe una funcionalidad compleja, con sus propios ciclos de vida de adquisición, desarrollo y evolución, y capacidades operacionales, vulnerabilidades y restricciones singulares. Estos nodos están diseñados para soportar Conjuntos de Flujos locales que se ocupan de las capacidades operacionales de obtención y mantenimiento. Pero también están diseñados para participar en las operaciones centradas en red para cumplir con los objetivos de la misión. En la parte central de la ilustración de la Figura 87, se ha definido un Conjunto de Flujos que soporta las operaciones de la misión recorriendo e integrando las capacidades de los componentes de red dentro de las capacidades globales coherentes. Por ejemplo, la integración de los flujos del sensor en el Conjunto de Flujos combina las salidas de los sensores en capas en una valorización abarcativa de las oportunidades de ataque. La integración de los flujos de disparo selecciona y coordina los elementos de ataque, y los flujos de valorización del daño integran los resultados de los sensores de reportes de ataque. Estos Conjuntos de Flujos centrados en red combinan las capacidades de los sistemas individuales para alcanzar los objetivos de la misión.

Por lo tanto, la tarea de desarrollar un sistema centrado en red es la de definir Conjuntos de Flujos para los sistemas particulares que soporten de modo transparente los Conjuntos de Flujos de la misión. Y los Conjuntos de Flujos deben estar diseñados para integrar estos sistemas dentro de capacidades coherentes. En este rol, los Conjuntos de Flujos son la principal especificación

de las capacidades de red, y los sistemas dentro de la red deben estar diseñados para soportarlos.

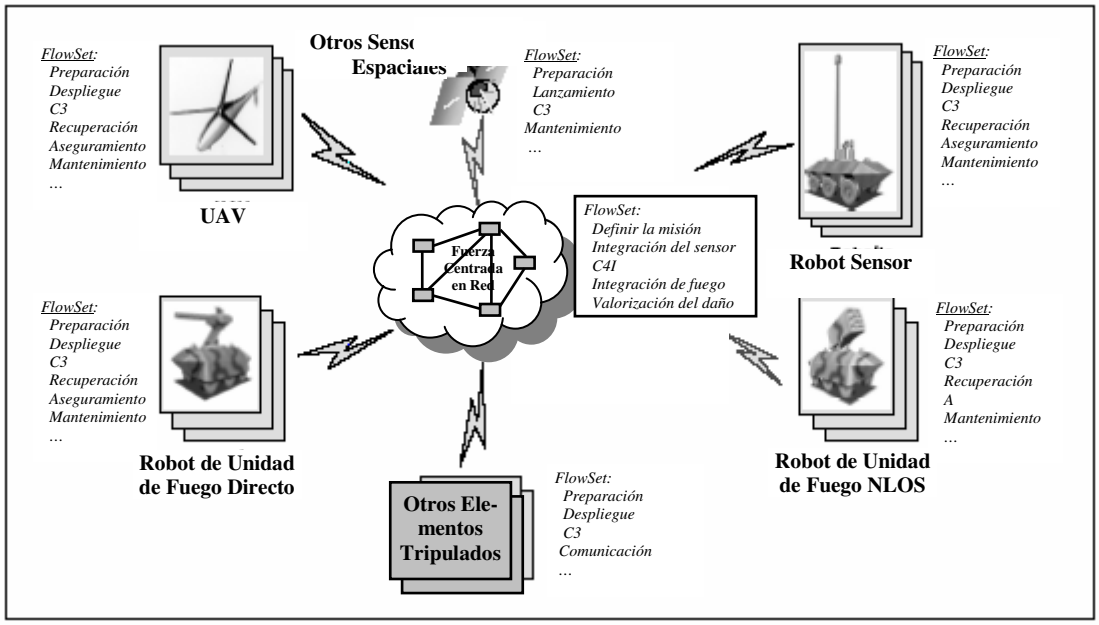


Figura 87. Conjuntos de Flujos para el Sistema de Combate del Futuro.

A5.2.5 Seguridad del flujo. Análisis de supervivencia

Los flujos representan el procesamiento extremo-a-extremo, es decir, los cruces desde un usuario (por ejemplo, una persona o un flujo) a través de la red y de regreso al usuario. Muchos sistemas y dominios pueden ser visitados durante estos cruces, cada cual con sus propias políticas y capacidades de seguridad y supervivencia. Esta situación se describe en la Figura 88 para el ejemplo de la transacción de compra de combustible. Esta transacción extremo-a-extremo debe ser segura y con capacidad de supervivencia. Los atributos de calidad para estas propiedades se pueden definir para el flujo en su conjunto, como requerimientos de *benchmark* que todos los sistemas y dominios atravesados deben proporcionar. La seguridad y la supervivencia se pueden caracterizar con respecto a la propagación del flujo, y sólo son tan buenas como el componente de menor capacidad. Algunos flujos deben ser más seguros y con mayor capacidad de supervivencia que otros, y los fabricantes [Ellison 99b, Ellison 99c, Linger 00, Mead 00a] y proveedores de servicio pueden ofrecer un espectro de garantías de calidad de servicio. No obstante, resulta frecuente que los flujos deban negociar por servicios y atributos de calidad en tiempo real. Este es el modelo básico FSQ, en el que los requerimientos de atributo asociados con los flujos son compatibilizados dinámicamente con las capacidades de red.

Las *Estructuras de Flujos* son un elemento importante para el *Survivable System Analysis* -SSA- la técnica desarrollada por el SEI CERT Coordination Center. SSA es un proceso estructurado

de ingeniería orientado a mejorar las características de supervivencia de sistemas nuevos o existentes. El mismo ha estado siendo aplicado a un número de sistemas gubernamentales y comerciales con buenos resultados. El proceso SSA identifica los flujos del sistema esenciales para la misión, esto es, los flujos que deben estar disponibles a pesar del ambiente de amenaza y del estado de compromiso. Estos usos de flujos de servicio son trazados a lo largo de la arquitectura del sistema para poner de manifiesto los componentes objeto de compromiso. Con esta información resulta posible identificar los puntos vulnerables que son tanto esenciales como pasibles de compromiso, seguido del análisis de supervivencia para el mejoramiento de la estrategia de resistencia, reconocimiento y recuperación dentro de la arquitectura del sistema.

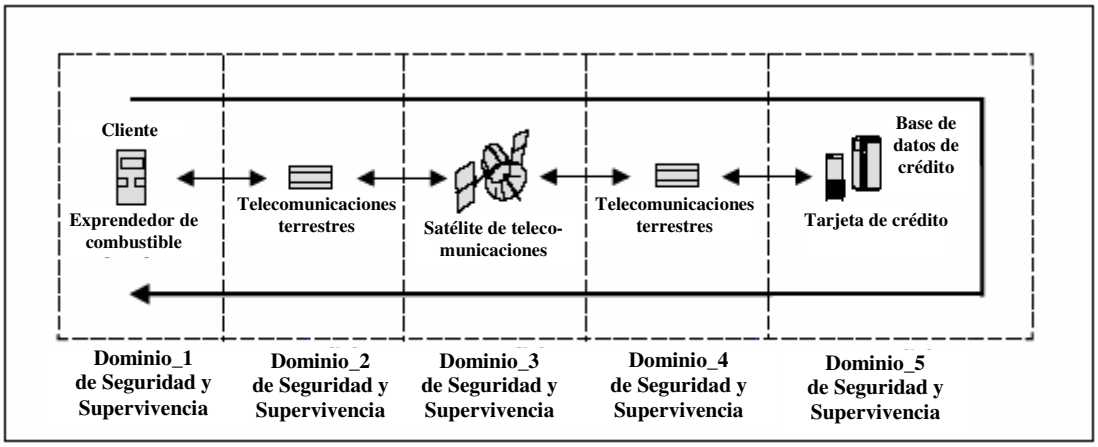


Figura 88. Seguridad y supervivencia de flujos que atraviesan dominios.

Estas seis operaciones de ingeniería ilustran algunos usos básicos de los conceptos y técnicas de la Estructura de Flujos. Operaciones adicionales incluyen la derivación de estructuras lógicas de red a partir de los Conjuntos de Flujos, y la definición de modelos de utilización estadísticos de los sistemas en red para la generación de casos de testeo basados en los patrones y probabilidades de utilización de los Conjuntos de Flujos.

A5.3 Un ejemplo de CQA

Se considera la solicitud de flujo sencilla ilustrada en la Figura 89. El usuario ha solicitado una instancia de flujo que consiste en la ejecución de una instancia del servicio A seguida por la ejecución de una instancia del servicio B. El CQA Q_1 contiene la solicitud de flujo. Q_1 está definido como la fiabilidad pronosticada, representada por un número real perteneciente al intervalo [0,1], y se la interpreta como la probabilidad de que una ejecución individual no falle.
Se especifican las siguientes restricciones CQA:

- Restricciones a nivel de servicio: $Q_1 > 0{,}97$ para la instancia de servicio A; y $Q_1 > 0{,}96$ para la instancia de servicio B

- Restricción a nivel de flujo: $Q_1 > 0{,}94$ para la instancia de flujo completo

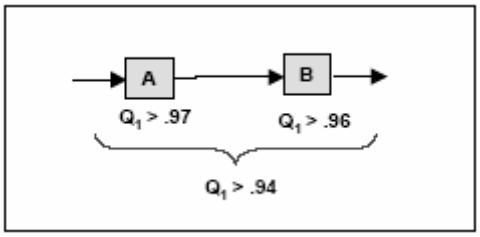


Figura 89: Solicitud de flujo restringida sencilla.

Representando las conexiones como los servicios *Con1*, *Con2*, y *Con3*, se asume que el flujo candidato está identificado con los siguientes pronósticos CQA:

$Q_1 = 0{,}99$ para *Con1*; $Q_1 = 0{,}98$ para A y *Con2*; $Q_2 = 0{,}97$ para B y *Con3*.

Estos valores CQA pronosticados satisfacen todas las restricciones a nivel de servicio especificadas por el usuario. En consecuencia, el próximo paso es determinar si el flujo candidato satisface cada restricción a nivel de flujo.

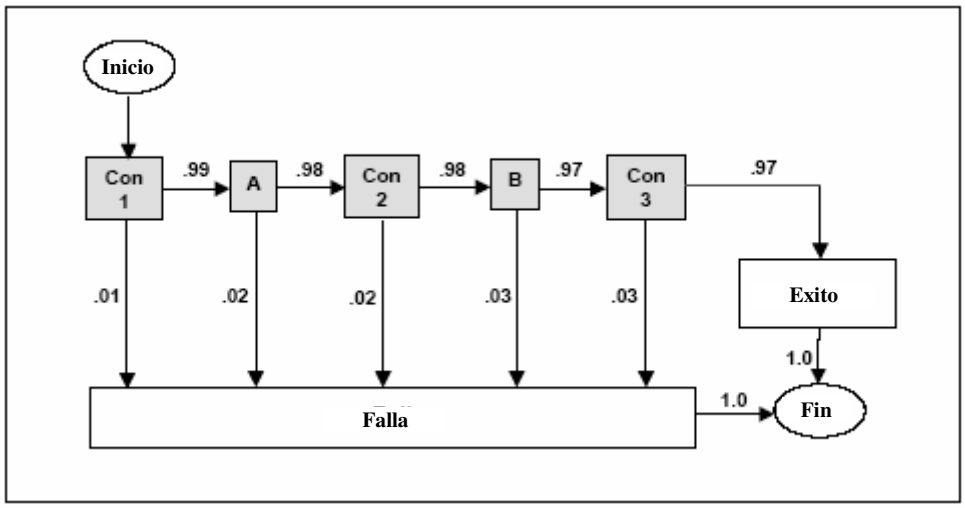


Figura 90. Diagrama de estado para el análisis de Q1.

La Figura 90 ilustra el diagrama de estado utilizado para analizar la restricción CQA a nivel de flujo $Q_1 = 0{,}95$ para el flujo candidato. Cada servicio se representa por un estado con dos arcos de partida: falla o satisface la restricción CQA y continúa con el siguiente servicio. Se continúa si el servicio satisface la restricción CQA especificada; falla si el servicio no puede satisfacer la restricción CQA. Las probabilidades asociadas con los arcos de partida para cada servicio suman hasta 1,0. En este ejemplo, si se asume el cumplimiento de la independencia de las probabilidades de cada servicio, la probabilidad de que el flujo candidato se ejecute sin falla es el producto de los valores Q_1 pronosticados para cada servicio. En el nivel de flujo, el Q_1 pronosticado

se calcula para este flujo candidato de la siguiente manera:

Q_1 pronosticado = 0.99 * 0.98 * 0.98 * 0.97* 0.97 = 0.89

En consecuencia, el flujo candidato no satisface la restricción a nivel de flujo de $Q_1 > 0{,}94$.

ANEXO 6 – EJEMPLOS DE APLICACIÓN DE ÁRBOLES DE ATAQUE

A6.1 Ejemplos de empleo de Patrones de Ataque

A lo largo de la pasada década, la forma más común de vulnerabilidad de la seguridad ha sido el incorrecto manejo de los *buffer overflows* por parte de los programas de computación [Cowan 00]. Como se muestra en la Figura 91, cuando un programa es invocado se agrega un registro de activación a la pila de ejecución del sistema. Cada registro de activación contiene la dirección de retorno cuando el programa termina y cualquier variable local y *buffers*. En ciertos programas, una entrada de usuario excesivamente grande puede provocar el desbordamiento del *buffer* interno. Como se muestra, el desbordamiento del *buffer* puede sobre-escribir las variables locales, el puntero de retorno, y otras porciones de la memoria adyacentes. Por ello, un atacante puede construir una entrada de usuario que modifique el puntero de retorno para que regrese a un código malicioso elegido por el atacante. Este código malicioso se ejecuta con el privilegio del programa original. Si el programa ejecuta con el privilegio del administrador, que es un caso frecuente, el atacante posee prácticamente el control completo del sistema. Este patrón de ataque se captura de la siguiente manera:

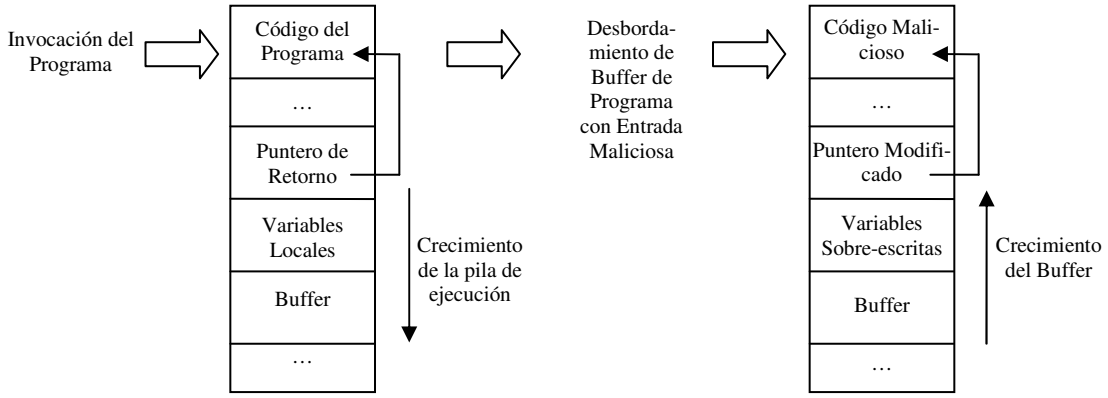


Figura 91. Ataque de Desbordamiento de Buffer.

Patrón de Ataque de Desbordamiento de Buffer*:*

Objetivo: Abusar la vulnerabilidad de Desbordamiento de Buffer para realizar una función maliciosa sobre le sistema objetivo.

Pre-condición: El atacante puede ejecutar ciertos programas sobre el sistema objetivo.

Ataque:

AND 1. Identificar en el sistema objetivo un programa ejecutable susceptible de la vulnerabilidad de desbordamiento de buffer.

2. Identificar el código que realizará la función maliciosa cuando se ejecute con el

privilegio del programa.

3. Construir el valor de entrada que forzará a que el código se encuentre en el espacio de direcciones del programa.
4. Ejecutar el programa de tal manera que lo haga saltar a la dirección en la que reside el código.

Post-condición: El sistema objetivo ejecuta la función maliciosa.

El Ataque de Desbordamiento de Buffer pone en evidencia una manera en la que un atacante abusa con medios maliciosos de la confianza que deposita un programa en la entrada del usuario. Este es un ejemplo de una clase más general de ataque denominado Ataques de Validación de Entrada (*Input Validation Attacks*): si el programa hubiera validado la entrada del usuario, tal vez truncándola apropiadamente, el programa no hubiera sido vulnerable al ataque. Otro ejemplo dentro de esta clase es el Ataque de Operador Inesperado (*Unexpected Operator Attack*). En lugar de ser vulnerable a los valores de entrada excesivamente largos, los programas susceptibles a la vulnerabilidad de Operador Inesperado simplemente no prevén que ciertos operadores serán incluidos en la entrada. Por ejemplo, el programa *p* de la Figura 92 aguarda un nombre de programa que será pasado como entrada de tal manera que el programa use los datos contenidos en el mismo con alguna finalidad. La vulnerabilidad del programa es abusada cuando un atacante adosa un nombre de archivo de entrada con un operador con estructura de comando (";" en este ejemplo) y un comando malicioso (remover todos los archivos del directorio corriente y de los directorios contenidos en él). El patrón asociado con este ataque es similar en formato al Patrón de Ataque de Desbordamiento de Buffer.

Patrón de Ataque de Operador Inesperado*:*

Objetivo: Abusar la vulnerabilidad de Operador Inesperado para realizar una función maliciosa sobre le sistema objetivo.

Pre-condición: El atacante puede ejecutar ciertos programas sobre el sistema objetivo.

Ataque:

AND 1. Identificar en el sistema objetivo un programa ejecutable susceptible de la vulnerabilidad de operador inesperado.

2. Identificar el operador (inesperado) que permite la composición de llamadas al sistema con el privilegio del programa.
3. Identificar la llamada al sistema que podría realizar la función maliciosa cuando se ejecute con el privilegio del programa.
4. Construir la entrada inesperado componiendo el valor de la entrada legal con la llamada al sistema que emplea el operador inesperado.
5. Ejecutar el programa sobre el sistema objetivo con la entrada inesperada.

Post-condición: El sistema objetivo ejecuta la función maliciosa.

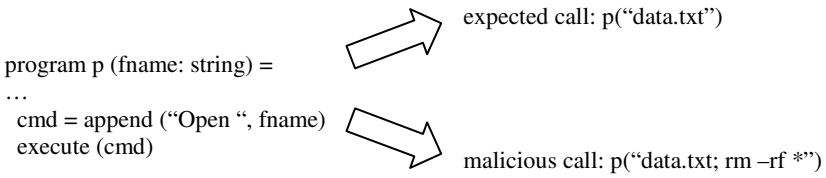


Figura 92. Ataque de Operador Inesperado.

Los patrones de ataque pueden existir en una variedad de niveles y no necesariamente conducen a un compromiso directo de la información o a una denegación de servicio. Simplemente pueden proporcionarle al atacante información que éste puede requerir para alcanzar su objetivo. Por ejemplo, descubrir los controles de acceso que son aplicados por un firewall resulta esencial para determinar de qué manera tomar el control de los sistemas que se encuentran más allá del firewall:

Patrón de Ataque de Descubrimiento del Control de Acceso*:*

Objetivo: Identificar los controles de acceso del firewall.

Pre-condición: El atacante conoce la dirección IP del firewall.

Ataque:

OR 1. Detectar puertos específicos por omisión en estado listening.
2. Sondear puertos para detectar aquéllos en estado listening.
3. Sondear furtivamente para detectar puertos en estado listening.

OR 1. Aleatorizar el objetivo del sondeo.
2. Aleatorizar la fuente del sondeo.
3. Sondear sin apuntar al host objetivo.

Post-condición: El atacante conoce los controles de acceso del firewall.

Otros patrones pueden ayudar a satisfacer las pre-condiciones de patrones ya especificados. El siguiente patrón ayuda a satisfacer la pre-condición del último patrón:

Patrón de Ataque de Descubrimiento de la Dirección IP:

Objetivo: Identificar la dirección IP del firewall objetivo.

Pre-condición: El atacante conoce el nombre de dominio del objetivo.

Ataque:

OR 1. Interrogar al Servidor del Nombre de Dominio.
2. Tracear la ruta a través del firewall hacia el servidor Web objetivo.
3. Sondear la dirección IP del firewall.

Post-condición: El atacante conoce la dirección IP del firewall objetivo.

A6.2 Ejemplos de empleo de Perfiles de Ataque

El Patrón de Ataque de Desbordamiento de Buffer es un miembro de este perfil y puede ser descripto nuevamente en términos de las variantes de la siguiente manera:

Patrón de Ataque de Desbordamiento de Buffer:

Objetivo: Abusar de la vulnerabilidad de desbordamiento de buffer a los fines de realizar una *función maliciosa* sobre el *Sistema*.

Pre-condición: El *Atacante* puede ejecutar ciertos programas en el *Sistema*.

Ataque:

AND 1. Identificar en el *Sistema* un programa ejecutable susceptible de la vulnerabilidad de desbordamiento de buffer.

2. Identificar el código que realizará la *función maliciosa* cuando el mismo se ejecute con el privilegio del programa.
3. Construir el valor de entrada que forzará que el código se ubique en el espacio de la dirección del programa.
4. Ejecutar el programa de tal manera que realice el salto hasta la dirección en la que reside el código.

Post-condición: El *sistema* realiza la *función maliciosa*.

Observar que las variantes del perfil *sistema* y *atacante* aparecen en el ejemplo anterior. Otra variante, *función maliciosa*, no aparece como un variante del modelo de referencia del perfil. La frase subrayada se encuentra definida en el glosario del perfil:

Vulnerabilidad de desbordamiento de buffer: una debilidad dentro de un programa que, cuando se lo ejecuta con valores de entrada excesivamente largos, hace que un buffer interno se desborde sobre-escribiendo porciones de una pila en ejecución y memoria adyacente.

ANEXO 7 – EJEMPLO DE REQUERIMIENTOS DE CALIDAD

Para asegurar que la discusión teórica resulte más específica, se consideran aquellos criterios de calidad asociados con el sub-factor integridad del factor de calidad seguridad. Los *tipos* de criterios de calidad que describen la integridad podrían ser los siguientes:

- Proteger las Transmisiones contra la Corrupción
- Detectar la Corrupción de los Datos Transmitidos
- Reaccionar ante la Corrupción de los Datos Transmitidos
- Proteger los Datos Almacenados *Online* contra la Corrupción
- Detectar la Corrupción de los Datos Almacenados *Online*
- Reaccionar ante la Corrupción de los Datos Almacenados *Online*
- Asegurar la Apropiada Restauración de los Datos y el Software en Caso de Corrupción
- Proteger los Componentes de Hardware contra la Corrupción
- Proteger los Componentes de Software contra la Corrupción
- Proteger los Componentes de Personal contra la Corrupción

Un ejemplo de una versión parametrizada del primer tipo de criterio de calidad anterior se podría establecer de la siguiente manera:

- "El A protege las transmisiones B sobre las redes C de la corrupción D por parte de los ataques E (o los atacantes E)," donde los parámetros precedentes se pueden reemplazar de la siguiente forma:
    - A se puede reemplazar por: negocio, centro, aplicación o componente.
    - B se puede reemplazar por: confidencial del personal, del negocio, o clasificada o todas.
    - C se puede reemplazar por: todas, públicas o internas.
    - D se puede reemplazar por: creación, modificación, borrado, repetición, o todas.
    - E se puede reemplazar por: todos, sofisticados o no-sofisticados.[46]

Por lo tanto, un *criterio de calidad* específico referido a la integridad para la protección de las *transmisiones* contra la corrupción se podría escribir de la siguiente manera: "La aplicación protege a las transmisiones de todo el personal sobre todas las redes públicas frente a todos los tipos de corrupción por parte de ataques no-sofisticados". Un ejemplo de un *criterio de calidad* específico referido a la protección de datos *almacenados online* contra la corrupción podría ser,

[46] Estos términos (por ejemplo, ataque no-sofisticado) deben estar oficialmente definidos dentro de algún tipo de glosario del proyecto. También existen otras posibles descomposiciones además de la sofisticación del ataque, incluidos ataques conocidos vs. Ataques desconocidos (por ejemplo, virus).

"La aplicación protege a la información de clientes almacenada, incluidos los balances contables, frente a la modificación no-autorizada por parte de ataques sofisticados".

En tanto los requerimientos funcionales tienden a ser binarios y se los especifica como todo o nada, los requerimientos de calidad se especifican utilizando una escala de medida. Por ejemplo, el desempeño sub-factor de calidad productividad (*throughput*) generalmente se especifica en términos de número de transacciones por unidad de tiempo, mientras que el sub-factor de calidad tiempo de respuesta generalmente se especifica en términos de segundos transcurridos. De manera similar, la escala de medición para el ejemplo de integridad de la sección anterior podría ser el siguiente:

- Porcentaje promedio de transmisiones protegidas por unidad de tiempo bajo condiciones dadas.
- Porcentaje promedio de transmisiones corrompidas por unidad de tiempo bajo condiciones dadas.

Si el criterio de calidad para la integridad de datos es, "La aplicación protege las transmisiones del personal sobre todas las redes públicas contra la corrupción mediante ataques no-sofisticados", entonces la métrica de calidad asociada podría ser "número promedio de corrupciones por hora". Estableciendo un nivel mínimo aceptable de esta métrica de calidad, obtenemos el siguiente requerimiento de integridad de datos: "Al menos el 99,99% del tiempo, la aplicación protegerá, por hora, las transmisiones del personal sobre todas las redes públicas contra la corrupción mediante ataques no-sofisticados".

Para hacer que el requerimiento precedente resulte verificable, resulta necesario requerir un porcentaje del tiempo en el que las transmisiones son exitosas (es decir, el test de seguridad para corromper la transmisión falla) como así también una carga de ataque específica. Esta información está destinada sólo a testear (verificar) y puede no corresponder con la futura carga de ataque real de la aplicación, lo que resulta muy difícil y casi imposible de estimar con precisión. Esta carga de ataque necesita incluir el nivel de esfuerzo del atacante ("una hora") como así también la indicación de la pericia y los recursos del atacante ("ataque no-sofisticado") que podría resultar necesario para que el ataque resulte exitoso.

REFERENCIAS

Los URLs son válidos a Diciembre del 2003.

[Albert 02] Albert, C. and Brownsword, L. *Evolutionary Process for Integrating COTS-Based Systems (EPIC)*, CMU/SEI-2002-TR-005, Software Engineering Institute, 2000 http://www.sei.cmu.edu/publications/documents/02.reports/02tr005.html.

[Alberts 03] Alberts, Christopher & Dorofee, Audrey. *Managing Information Security Risks: The Octave™ Approach*. Boston, MA: Addison Wesley, 2003.

[Allen 00] Allen, J.; Christie, A.; Fithen, W.; McHugh, J.; Pickel, J.; & Stoner, E. *State of the Practice of Intrusion Detection Technologies* (CMU/SEI-99-TR-028, ADA375846). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2000. http://www.sei.cmu.edu/publications/documents/99.reports/99tr028/99tr028abstract.html

[Allen 01] Allen, Julia H. *The CERT Guide to System and Network Security Practices*. Boston, M.A.: Addison Wesley, 2001.

[Anderson 93] Anderson, R.*Why Cryptosystems Fail*, in Proc. 1st Conf. On Computer and Communications Security, 1993.

[Anderson 97] Anderson, R. H.; Hearn, A. C.; & Hundley, R. O. *RAND Studies of Cyberspace Security Issues and the Concept of a U.S. Minimum Essential Information Infrastructure*. 1997. http://www.cert.org/research/isw97_hypertext/all_the_papers/no1.html

[Anderson 99] Anderson, R. H.; Feldman, P. M.; Gerwehr, S.; Houghton, B. K.; Mesic, R.; Pinder, J.; Rothenberg, J.; & Chiesa, J. R. *Securing the U.S. Defense Information Infrastructure: A Proposed Approach* (RAND Report MR-993-OSD/NSA/DARPA). Santa Monica, CA: RAND Corporation, 1999. http://www.rand.org/publications/MR/MR993

[Anderson 01] Anderson, Ross. *Security Engineering: A Guide to Building Dependable Distributed Systems*. New York, NY: Wiley Computer Publishing, 2001.

[Arbaugh 00] Arbaugh, W.A., W.L. Fithen, and J. McHugh. *Windows of Vulnerability: A Case Study Analysis*, IEEE Computer, Vol. 33, No. 12, Dec. 2000.

[Bachmann 02] Bachmann, F.; Bass, L.; & Klein, M. *Illuminating the Fundamental Contributors to Software Architecture Quality* (CMU/SEI-2002-TR-025). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2002. http://www.sei.cmu.edu/publications/documents/02.reports/02tr025.html

[Barbacci 00] Barbacci, Mario R.; Ellison, Robert J.; Weinstock, Charles B.; & Wood, William G. *Quality Attribute Workshop Participant's Handbook* (CMU/SEI-2000-SR-001). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2000. http://www.sei.cmu.edu/publications/documents/00.reports/00sr001.html

[Basili 01] Basili, V. R. & Boehm, B. *COTS-Based Systems Top 10 List,* IEEE Software 34, 5 (May 2001): 91-93.

[Bass 01] Bass, L.; Klein, M.; & Bachmann, F. *Quality Attribute Design: Primitives and the Attribute Driven Design Method,*. 4th Conference on Product Family Engineering. Bilbao, Spain, 4 October 2001. http://www.sei.cmu.edu/plp/bilbao_paper.pdf

[Birman 96] Birman, Kenneth P. *Building Secure and Reliable Network Application,* Greenwich, Conn: Manning, 1996.

[Boehm 76] Boehm, Barry; Brown, J. R.: & Lipow, M. *Quantitative Evaluation of Software Quality*, 592-605. Proceedings of the 2nd International Conference on Software Engineering, San Francisco, CA, Oct. 13-15, 1976. New York, NY: IEEE Computer Society, 1976.

[Boehm 88] Boehm, B. *A Spiral Model of Software Development and Enhancement,* IEEE Communications 21, 5 (May 1988): 61-72.

[Boehm 89] Boehm, B. W. *Software Risk Management,* IEEE Computer Society Press, 1989.

[Boehm 00] Boehm, B., Abts, C. and Bailey Clark, E. *COCOTS: a COTS Software Integration Cost Model*, Proceedings ESCOM-SCOPE 2000 Conference, April 2000.

[Brown 98] Brown, A. And Wallnau, K. *The Current State of CB SE*. IEEE Software, Vol. 15, Nº 5, Sept/Oct, 1998.

[Brownsword 00] Brownsword, Lisa; Oberndorf, Tricia; & Sledge, Carol A. *Developing New Processes for COTS-based Systems*, IEEE Software 17, 4 (July/August 2000): 48-55.

[Carrol 99] Carrol, John M. *Five Reasons for Scenario-Based Design*, Proceedings of the Thirty-Second Annual Hawaii International Conference on Systems Sciences. Maui, Hawaii, Jan. 5-8, 1999. Los Alamitos, CA: IEEE Computer Society Press, 1999.

[CCIMB 99] Common Criteria Implementation Board. , (CCIMB-99-031) [online] August 1999). http://csrc.ncsl.nist.gov/cc/ccv20/ccv2list.htm

[CERT 01] CERT Coordination Center. *Managing the Threat of Denial-of-Service Attacks*. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, October 2001. http://www.cert.org/archive/pdf/Managing_DoS.pdf

[CERT 02] CERT Coordination Center. *Overview of Attack Trends*. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2002. http://www.cert.org/archive/pdf/attack_trends.pdf

[Chung 93] Chung, Kyungwha Lawrence. *Representing and Using Non-Functional Requirements: A Process-Oriented Approach* (DKBSTR-93-1, Ph D dissertation). Toronto, Canada: Department of Computer Science, University of Toronto, 1993.

[Chung 00] Chung, Kyungwha Lawrence; Nixon, Brian A.; Yu, Eric; & Mylopoulos, John, *Non-Functional Requirements in Software Engineering*. Norwell, MA: Kluwer Academic Publishers, 2000.

[Clements 02] Clements, P.; Bachmann, F.; Bass, L.; Garlan, D.; Ivers, J.; Little, R.; Nord, R.; & Stafford, J. *Documenting Software Architectures: Views and Beyond*. Boston, MA: Addison Wesley Longman, 2002.

[CNSS 03] Committee on National Security Systems (CNSS). *National Information Assurance (IA) Glossary* (CNSS Instruction No. 4009). Fort Meade, Maryland: Committee on National Security Systems (CNSS), National Security Agency (NSA), May 2003.

[Cowan 00] Cowan, C.; Wagle, P.; Pu, C.; Beattie, S.; & Walpole, J. *Buffer Overflows: Attacks and Defenses for the Vulnerability of the Decade,* Proceedings of the DARPA Information Survivability Conference and Expo (DISCEX). Hilton Head, SC, January 25-27, 2000. Los Alamitos, CA: IEEE Computer Society, 2000.

[Coyle 00] Coyle, G. *Qualitative and Quantitative Modeling in System Dynamics: Some Research Questions*, System Dynamics Review 16, 3 (Fall 2000): 225-244.

[Coyle 96] Coyle, G. *System Dynamics Modeling*, New York: Chapman & Hall, 1996.

[Cybenko 02] Cybenko, G.; Giani, A.; & Thompson, P. *Cognitive Hacking: A Battle for the Mind* IEEE Computer 35, 8 (August 2002): 50-63.

[Davis 93] Davis, Alan M. *Software Requirements: Objects, Functions, and States*, Englewood Cliffs, NJ: Prentice Hall, 1993.

[DITSCAP 99] U.S. Department of Defense. *DoD Information Technology Security Certification and Accreditation Process (DITSCAP).* DoD Instruction 5200.40, 30 November 1999.
http://www.sabi.org/history.htm

[DoD 85] Department of Defense. *Department of Defense Trusted Computer System Evaluation Criteria,* DoD 5200.28-STD. National Computer Security Center, Department of Defense Computer Security Center, 1985.

[DoD 02] U.S. Department of Defense. *Information Assurance (IA) Directive* 8500.1, October 22, 2002.
http://www.dtic.mil/whs/directives/corres/pdf/d85001_102402/d85001p.pdf

[Ebert 97] Ebert, C. “Dealing with Nonfunctional Requirements in Large Software Systems”. 367-395. *Annals of Software Engineering 3*, September 1997.

[Ellison 98a] Ellison, Robert; Fisher, David; Linger, Richard; Lipson, Howard; Longstaff, Thomas; & Mead, Nancy. *A Survivable Network Analysis Method*, Proceedings of the 1998 IEEE Information Survivability Workshop. Orlando, Florida, Oct. 28-30, 1998. Los Alamitos, CA: IEEE Computer Society, 1998.
http://www.cert.org/research/isw.html

[Ellison 98b] Ellison, R.; Linger, R.; Longstaff, T.; Mead, N. *Case Study in Survivable Network System Analysis* (CMU/SEI-98-TR-014, ADA355070). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1998. http://www.sei.cmu.edu/publications/documents/98.reports/98tr014/98tr014abstract.html

[Ellison 99a] Ellison, R. J.; Fisher, D. A.; Linger, R. C.; Lipson, H. F.; Longstaff, T. A.; & Mead, N. R. *Survivable Systems: An Emerging Discipline*, 93-116. Proceedings of the 11th Canadian Information Technology Security Symposium (CITSS). Ottawa, Ontario, Canada, May 10-14, 1999. Ottawa, Ontario, Canada: Communications Security Establishment, 1999. http://www.sei.cmu.edu/publications/documents/97.reports/97tr013/97tr013abstract.html

[Ellison 99b] Ellison, R.; Fisher, D.; Linger, R.; Lipson, H.; Longstaff, T.; & Mead, N. *Survivable Network System Analysis: A Case Study,* IEEE Software 16, 4 (July/August, 1999): 70-77.

[Ellison 99c] Ellison, R.; Fisher, D.; Linger, R.; Lipson, H.; Longstaff, T.; & Mead, N. *Survivability: Protecting Your Critical Systems*, IEEE Internet Computing 3, 6 (November/December, 1999).

[Ellison 03] Ellison, Robert J. & Moore, Andrew P. *Trustworthy Refinement Through Intrusion-Aware Design (TRIAD)* (CMU/SEI-2003-TR-002). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, March 2003. http://www.sei.cmu.edu/publications/documents/03.reports/03tr002.html

[Firesmith 02] Firesmith, Donald & Henderson-Sellers, Brian. *The OPEN Process Framework.* London, England: Addison-Wesley, 2002.

[Firesmith 03a] Firesmith, Donald G. *Using Quality Models to Engineer Quality Requirements*, Journal of Object Technology (JOT) 2, 5 (September/October 2003): 67-75. Zurich, Switzerland: Swiss Federal Institute of Technology (ETH). http://www.jot.fm/issues/issue_2003_09/column6

[Firesmith 03b] Firesmith, Donald G. *Analyzing and Specifying Reusable Security Requirements*, 7-11. Requirements Engineering 2003 Requirements for High Assurance Systems (RHAS) Workshop Proceedings, Monterey, CA, Sept. 9, 2003. Washington, D.C.: IEEE Computer Society, 2003.

[Firesmith 03c] Firesmith, Donald G. *Specifying Reusable Security Requirements*, Journal of Object Technology (JOT) 3, 1 (January/February 2004): 61-75. http://www.jot.fm/issues/issue_2004_01/column6

[Firesmith 03d] Firesmith, Donald G. *Firesmith's OPEN Process Framework Website*. 2003 http://www.donald-firesmith.com

[Firesmith 03e] Firesmith, Donald G. "*Common Concepts Underlying Safety, Security, and Survivability Engineering"*. Software Engineering Institute. Carnegie Mellon University. December 2003. www.sei.cmu.edu/publications/ documents/03.reports/03tn033.html

[Fisher 99] Fisher, D. A. & Lipson, H. J. *Emergent Algorithms: A New Method for Enhancing Survivability in Unbounded Systems*, Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1999. http://www.cert.org/archive/html/emergentalgor.html

[Forrester 61] Forrester, J. W. *Industrial Dynamics*. Republished by Productivity Press, Portland, OR. Cambridge, MA: MIT Press, 1961.

[Froscher 98] Froscher, J. & Kang, M. *A Client-Server Architecture Supporting MLS Interoperability with COTS Components*, MILCOM '97. Conference Proceedings. Monterey, CA, November 1997. Piscataway, NJ: IEEE Service Center, 1997

[Glinz 00] Glinz, M. *Problems and Deficiencies of UML as a Requirements Specification Language*. Proceedings of the Tenth International Workshop on Software Specification and Design, IEEE Computer Society Press, Los Alamitos, CA, 2000, pp. 11-22.

[Gokhale 98] Gokhale, S. & Trivedi, K. *Dependency Characterization in Pathbased Approaches to Architecture-based Software Reliability Prediction,* Proceedings of the 1998 IEEE Workshop on Application Specific Software Engineering and Technology. Richardson, Texas, March 26-28, 1998. Los Alamitos, CA: IEEE Computer Society Press, 1998.

[Haeckel 99] Haeckel, S. *Adaptive Enterprise: Creating and Leading Sense-and-Respond Organizations*. Boston: Harvard Business School Press, 1999.

[Hamlet 01] Hamlet, D.; Mason, D.; & Woit, D. *Theory of Software Reliability Based on Components*,.Proceedings of the 23rd International Conference on Software Engineering (ICSE 2002). Toronto, Canada, May 2001. Los Alamitos, CA: IEEE Computer Society Press, 2001.

[Hassler 01] Hassler, V. *Security Fundamentals for E-Commerce*, Proceedings of the 1st Conference On Computer and Communications Security. Norwood, MA: Artech House, Inc., 2001.

[Hausler 90] Hausler, P.; Pleszkoch, M.; Linger, R.; & Hevner, A. *Using Function Abstraction to Understand Program Behavior*, IEEE Software 7, 1 (January 1990): 55-63.

[Herrmann 99] Herrmann, Debra. *Software Safety and Reliability*, Los Alamitos, CA: IEEE Computer Society, 1999.

[Hevner 01] Hevner, A.; Linger, R.; Sobel, A.; & Walton, G. *Specifying Large-Scale, Adaptive Systems with Flow-Service-Quality (FSQ) Objects*, Proceedings of the 10th OOPSLA Workshop on Behavioral Semantics. Tampa, Florida, October 2001.

[Hevner 02] Hevner, A.; Linger, R.; Sobel, A.; & Walton, G.. *The Flow-Service-Quality Framework: Unified Engineering for Large-Scale, Adaptive Systems*, Proceedings of the 35th Annual Hawaii International Conference on System Science (HICSS35). Hawaii, 2001. Los Alamitos, CA: IEEE Computer Society Press, 2002.

[Hissam 98] Hissam, S.; Carney, D.; & Plakosh, D. *SEI Monograph Series: DoD Security Needs and COTS-Based Systems* (monograph). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, September 1998.

[Hoffman 01] Hoffman, D. & Weiss, D. *Software Fundamentals: Collected Papersby David L. Parnas*. Upper Saddle River, NJ: Addison Wesley, 2001.

[Hughes 95] Hughes, Larry. *Actually Useful Internet Security Techniques*, Indianapolis, Indiana: New Riders, 1995.

[IATF 02] Information Assurance Technical Forum. *The Information Systems Security Engineering Process*, IATF Release 3.1, September 2002.

[ISO 00] International Standards Organization (ISO). *Software Engineering - Product Quality - Part 1: Quality Model*, ISO/IEC 9126-1, Quebec, Canada: ISO, 2000.

[Jacobson 92] Jacobson, I., Christerson, M., Jonsson, P. And Overgaad, G. *Object-Oriented Software Engineering: A Use Case Driven Approach.* Addison-Wesley Publishers, Reading, MA, 1992.

[Jacobson 99] Jacobson, Ivar; Booch, Grady; & Rumbaugh, James. *The Unified Software Development Process.* Boston, MA: Addison Wesley Longman, 1999.

[Joyner 01] Joyner, I. *Open Distributed Proccesing Unplugged*. http://homepages.tig.com.au/~ijoyner/ODPUnplugged.html, 2001.

[Kazman 98] R. Kazman, M. Klein, M. Barbacci, T. Longstaff, H. Lipson, and S.J. Carriere, *The Architecture Tradeoff Analysis Method,* Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems, August 1998, Monterey, Calif., IEEE Computer Society. http://www.sei.cmu.edu/activities/architecture/projects.html

[Kemmerer 91] Kemmerer, R.A. & Porras, P.A. *Covert Flow Trees: A Visual Approach to Analyzing Covert Storage Channels*, IEEE Transactions on Software Engineering 17, 11 (November 1991): 1166-1185.

[Knight 00a] Knight, John C. & Sullivan, Kevin J. *On the Definition of Survivability* (Technical Report CS-TR-33-00). Charlottesville, VA: University of Virginia, Department of Computer Science, 2000.

[Knight 00b] Knight, J. C.; Sullivan, K. J.; Elder, M. C.; & Wang, C. *Survivability Architectures: Issues and Approaches*, Proceedings of the 2000 DARPA Information Survivability Conference and Exposition (DISCEX 2000). Hilton Head, South Carolina, Jan. 25-27, 2000. Los Alamitos, CA: IEEE Computer Society, 2000.

[Knight 03] Knight, John C.; Strunk, Elisabeth A.; & Sullivan, Kevin J. *Towards a Rigorous Definition of Information System Survivability*, 78. Proceedings of DISCEX 2003. Washington, D.C., April 2003.

[Kontio 96] Kontio, J. *A Case Study in Applying a Systematic Method for COTS Selection*, Proceedings of the 18th ICSE, IEEE, 1996.

[Krishnamurthy 02] Krishnamurthy, S. & Mathru, A. *On the Estimation of Reliability of a Software System Using Reliabilities of its Components*, Proceedings of Eighth International Symposium on Software Reliability Engineering (ISSRE '97). Albuquerque, New Mexico, November 1997. Los Alamitos, CA: IEEE Computer Society Press, 2002.

[Lee 89] Lee, P. *Bayesian Statistics: an Introduction*. New York: Oxford University Press, 1989.

[Leveson 95] Leveson, N. G. *Safeware: System Safety and Computers*, New York, New York: Addison-Wesley, 1995.

[Leymann 00a] Leymann, F. & Roller, D. *Production Workflow: Concepts and Techniques*. Upper Saddle River, NJ: Prentice-Hall PTR, 2000.

[Leymann 00b] Leymann, F. and Roller, D. *Production Workflow: Concepts and Thecniques*. Prentice-Hall PTR, Upper Saddle River, NJ, 2000.

[Lindqvist 98] Lindqvist, U. & Johnson, E. *A Map of Security Risks Associated with Using COTS*, IEEE Computer 31, 6 (June 1998): 60-66

[Linger 79] Linger, R.; Mills, H.; & Witt, B. *Structured Programming: Theoryand Practice*. Reading, MA: Addison-Wesley, 1979.

[Linger 98] R. C. Linger, N. R. Mead, and H. F. Lipson, *Requirements Definition for Survivable Network Systems*, Proceedings of the International Conference on Requirements Engineering, April 1998, IEEE Computer Society Press, p. 14-23.
http://www.cert.org/archive/pdf/icre.pdf

[Linger 99] Linger, Richard & Trammell, Carmen J. *Cleanroom Software Engineering Theory and Practice*, 351-372. Industrial Strength Formal Methods in Practice. Hinchey, Mike & Bowen, Jonathan, eds. London, UK: Springer-Verlag, 1999.

[Linger 00] Linger, R.; Ellison, R.; Longstaff, T.; & Mead, N. *The Survivability Imperative: Protecting Critical Systems*, Crosstalk 13, 10 (October 2000): 12-15.

[Lipson 99] Lipson, H. & Fisher, D. *Survivability – A New Technical and Business Perspective on Security*, Proceedings of the New Security Paradigms Workshop. Caledon Hills, Ontario, Canada, September 22-24, 1999, New York, NY: ACM.
http://www.cert.org/research/

[Loucopoulos 95] Loucopoulos, P. & Karakostas, V. *System Requirements Engineering*. New York, NY: McGraw Hill, 1995.

[MAFTIA 02] MAFTIA Partners. “Malicious- and Accidental-Fault Tolerance for Internet Applications.” IST Programme RTD Research Project IST-1999-11583.
http://www.newcastle.research.ec.org/maftia

[Maiden 98] Maiden, N. And Ncube, C. *Acquiring COTS Software Selection Requirements*, IEEE Software, March/April 1998.

[Maier 00] Maier, M. W. & Rechtin, E. *The Art of Systems Architecting*. Boca Raton, FL: CRC Press, 2000.

[Marmor-Squires 88] Marmor-Squires, A.B. and Rougeau, P.A. *Issues in Process Models and Integrated Environments for Trusted Systems Development*, Proceedings of the 11th National Computer Security Conference. October 1988.

[Marmor-Squires 89] Marmor-Squires, A., McHugh J., Branstad M., Danner B., Nagy L., Rougeau P., and Sterne D., *A Risk Driven Process Model for the Development of Trusted Systems*, Proceedings of the 1989 Computer Security Applications Conference. Tucson, AZ, December 1989. PP 184-192

[McDermid 91] McDermid, John. *Issues in Developing Software for Safety Critical Systems*, Reliability Engineering and Systems Safety 32, 1-2 (1991): 1-24.

[McDermott 99] McDermott, J. & Fox, C. *Using Abuse Case Models for Security*, Proceedings of the 15th Annual Computer Security Applications Conference. Phoenix, Arizona, Dec. 6-10, 1999. Los Alamitos, CA: IEEE Computer Society, 1999.
http://www.computer.org/proceedings/acsac/0346/0346toc.htm

[McHugh 84] McHugh, John. *Towards the Generation of Efficient Code From Verified Programs*. PhD Dissertation, The University of Texas at Austin, Austin, TX, 1984.

[McHugh 00] [McHugh, J.; Christie, A.; & Allen, J. *Defending Yourself: The Role of Intrusion Detection Systems,* IEEE Software 17, 5 (September/ October 2000): 42-51.

[McNamara 03] McNamara, Joel. *Secrets of Computer Espionage: Tactics and Countermeasures*. Indianapolis, IN: Wiley, 2003.

[Mead 00a] Mead, Nancy; Ellison, Robert; Richard, Linger; Longstaff, Thomas; & McHugh, John. *Survivable Network Analysis Method* (CMU/SEI-2000-TR-013, ADA383771). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2000.
http://www.sei.cmu.edu/publications/documents/00.reports/00tr013.html

[Mead 00c] Mead, Nancy; Linger, Richard; McHugh, John; & Lipson, Howard. *Managing Software Development for Survivable Systems*, Annals of Software Engineering 11, 1 (November 2001): 45-78.

[Mead 01] Mead, N. R.; Lipson, H. F.; & Sledge, C. A. *Towards Survivable COTS-Based Systems*, Cutter IT Journal 14, 2 (February 2001): 4-11.

[Mead 03] Mead, Nancy R. *Requirements Engineering for Survivable Systems* (CMU/SEI-2003-TN-013). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, September 2003.
http://www.sei.cmu.edu/publications/documents/03.reports/03tn013.html

[Mills 86] Mills, H.; Linger, R; & Hevner, A. *Principles of Information System Analysis and Design*. San Diego: Academic Press, 1986.

[Mills 88] Mills, H. *Stepwise Refinement and Verification in Box-Structured Systems*, IEEE Computer 21, 6 (June 1988): 23-36.

[Mills 92] Mills, H. D. *Certifying the Correctness of Software*, vol 2, 373-381. Proceedings of 25th Hawaii International Conference on System Sciences. Kauai, Hi., January 7-10, 1992. Los Alamitos, Calif.: IEEE Computer Society Press, 1992.

[Mills 02] Mills, H. & Linger, R. *Cleanroom Software Engineering. Wiley Encyclopedia of Software Engineering: Second Edition.* New York: Wiley, 2002.

[Moffett 03] Moffett, Jonathan D. & Nuseibeh , Bashar A. *A Framework for Security Requirements Engineering* (Report YCS 368). United Kingdom: Department of Computer Science, University of York, 20 August 2003.

[Moore 01a] Moore, A. P.; Ellison, R. J.; & Linger, R. C. *Attack Modeling for Information Security and Survivability* (CMU/SEI-2001-TN-001, ADA388771). Pittsburgh, PA: Software Engineering Institute,Carnegie Mellon University, 2001.
http://www.sei.cmu.edu/publications/documents/01.reports/01tn001.html

[Moore 01b] Moore, A. P. & Ellison, R. J. *Attack Modeling for Survivable System Analysis*, in Proceedings of the Information Systems Survivability Workshop, Dependable Systems and Networks Conference, Gothenburg, Sweden, July 2001.
http://www.cert.org/archive/pdf/intrusion-aware.pdf

[Mylopoulos 92] Mylopoulos, John; Chung, Lawrence; & Nixon, Brian. *Representing and Using Non-Functional Requirements: A Process-Oriented Approach*, IEEE Transactions on Software Engineering, Special Issue on Knowledge Representation and Reasoning in Software Development 18, 6 (June 1992): 483-497.

[NCSC 88] National Computer Security Center. *Glossary of Computer Security Terms*, NCSC-TG-004 Version 1, October 1988

[NCSC 91] National Computer Security Center. *A Guide to Understanding Identification and Authentication in Trusted Systems*, NCSCTG-017 Version 1, September 1991.

[Neumann 00] Neumann, P. G. *Practical Architectures for Survivable Systems and Networks*. Technical report, Computer Science Laboratory, SRI International, Menlo Park, California, 30 June 2000.
http://www.csl.sri.com/neumann/survivability.pdf

[Oberndorf 00] Oberndorf, Tricia; Brownsword, Lisa; & Sledge, Carol A. *An Activity Framework for COTS-Based Systems* (CMU/SEI-2000-TR-010, ADA383836). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2000.
http://www.sei.cmu.edu/publications/documents/00.reports/00tr010.html

[OPSEC 00] The Centre for Counterintelligency and Security Studies, *Intelligence Threat Handbook*. Greenbelt, MD: Interagency OPSEC Support Staff, June 2000.

[Parnas 86] D. L. Parnas and P. C. Clements, *A Rational Design Process: How and Why to Fake It*. IEEE Transactions on Software Engineering SE-12(2): 251-257, February 1986.

[Parnas 94] Parnas, D. & Wang, Y. *Simulating the Behavior of Software Modules by Trace Rewriting Systems*, IEEE Transactions on Software Engineering 19, 10 (October 1994): 750-759.

[Peltier 01] Peltier, Thomas R. *Information Security Risk Analysis*. Boca Raton, FL: CRC Press, 2001.

[Pleszkock 90] Pleszkoch, M.; Hausler, P.; Hevner, A.; & Linger, R. *Function-Theoretic Principles of Program Understanding,* Proceedings of the 23rd Annual Hawaii International Conference on System Science (HICSS23). Los Alamitos, CA: IEEE Computer Society Press, Los Alamitos, CA, 1990.

[Pleszkoch 02] Pleszkoch, M.; Linger, R.; Walton, G.; & Hevner, A. *Semantic Foundations for Flow Structures*. A publicar.

[Polen 99] Polen, S., Rose, L. and Philips, B. *Component Evaluation Process*. SPC98091-CMC, Software Productivity Consortium, 1999.

[Potts 95] Potts, C. *Using Schematic Scenarios to Understand User Needs,* 247-256. Proceedings of DIS'95—ACM Symposium on Designing Interactive Systems: Processes, Practices, Methods, & Techniques. Ann Arbor, Michigan, Aug. 23-25, 1995. New York: ACM Press, 1995.

[Power 00] Power, Richard. *Tangled Web: Tales of Digital Crime from the Shadows of Cyberspace*. Indianapolis, IN: QUE, 2000.

[Prakken 00] Prakken, B. *Information, Organization and Information Systems Design*, Norwell, MA: Kluwer Academic Publishers, 2000.

[Prowell 99] Prowell, S.J., C.J. Trammell, R.C. Linger, and J.H. Poore, *Cleanroom Software Engineering: Technology and Process*, Addison Wesley Longman, Inc., 1999.

[Ramachandran 02] Ramachandran, J. *Designing Security Architecture Solutions*. New York: John Wiley & Sons, 2002.

[Ramesh 97] Ramesh, B.; Stubbs, C.; Powers, T.; & Edwards, M. *Requirements Traceability – Theory and Practice*, Annals of Software Engineering, 3 (1997): 397-415.

[Ramesh 98] Ramesh, B. *Factors Influencing Requirements Traceability Practice*, Communications of the ACM 41, 12 (December 1998):37-44.

[Roman 85] Roman, Gruia-Catalin. *A Taxonomy of Current Issues in Requirements Engineering, IEEE Computer 18,* 4 (April 1985): 14-23.

[Royce 87] W. W. Royce, *Managing the Development of Large Software Systems,* Proceedings of the 9th International Conference on Software Engineering, Monterey, CA, IEEE Computer Society Press, Los Alamitos, CA, 1987.

[Royal 97] Royall, R. *Statistical Evidence: a Likelihood Paradigm*. New York: Chapman & Hall, 1997.

[Rushby 83] Rushby, J. M. & Randell, B. *A Distributed Secure System*, Proceedings of the IEEE Symposium on Security and Privacy, Oakland, California, April 25-27, 1983. Maryland: IEEE Computer Society Press, 1984.

[Salter 98] Salter, C., O. Saydjari, B. Schneier, J. Walner, *Toward a Secure System Engineering Methodology*, Proceedings Of New Security Paradigms Workshop. Charlottesville, Virginia, Sept. 22-25, 1998. New York: ACM Press, 1998.

[Saltzer 84] Saltzer, J. H.; Reed, D. P.; & Clark, D. D. *End-to-End Arguments in System Design*, ACM Transactions on Computer Systems 2, 4 (November 1984): 277-288.

[Schneider 99] Schneider, F. (ed.). *Trust in Cyberspace*. Washington, DC: National Academy Press, 1999.

[Schneier 99] Schneier, B., *Attack Trees: Modeling Security Threats*, Dr. Dobb's Journal, December 1999.

[Schneier 00a] Schneier, B. *Secrets and Lies: Digital Security in a Networked World*. New York, NY: Wiley, 2000.

[Schneier 00b] Schneier, B. *Closing the Window of Exposure: Reflections on the Future of Security*, Securityfocus.com, 2000. http://online.securityfocus.com/guest/3384

[SEI 02] Software Engineering Institute, *COTS Usage Risk Evaluation (CURE)*. October 2002, http://www.sei.cmu.edu/cbs/CURE_intro.html

[Shema 03] Shema, Mike. *Hack Notes: Web Security Portable Reference*. Emeryville, CA: McGraw Hill, 2003.

[Siegrist 88] Siegrist, K. *Reliability of Systems with Markov Transfer of Control,* IEEE Transactions on Software Engineering 14, 9 (September 1988): 1049-1053.

[Sikora 98] Sikora, R. & Shaw, M. *A Multi-Agent Framework for the Coordination and Integration of Information Systems*, Management Science 44, 11 (1998): S65-S78.

[Sindre 00] Sindre, G. & Opdahl, A. L. *Eliciting Security Requirements by Misuse Cases*, Proceedings of Conference on Technology of Object-Oriented Languages and Systems. Sydney, NSW, Australia, Nov. 20-23, 2000. Los Alamitos, CA: IEEE Computer Society Press, 2000.

[Sommerville 92] Sommerville, Ian. *Software Engineering*, fourth edition. Reading, Ma: Addison-Wesley, 1992.

[Soo Hoo 00] Soo Hoo, K. J. *How Much Is Enough? A Risk-Management Approach to Computer Security*. Report of the Consortium for Research on Information Security and Policy, Center for International Security and Cooperation, Stanford University, June 2000. http://ldml.stanford.edu/cisac/pdf/soohoo.pdf

[Schmidt 99] Schmidt, M. *The Evolution of Workflow Standars*. *IEEE Concurrency*. July-September, 1999, pp 44-52.

[Stavely 98] Stavely, Allan M. *Toward Zero-Defect Programming*. Boston, MA: Addison Wesley, 1998.

[Sterman 00] Sterman, J. D. *Business Dynamics: Systems Thinking and Modeling for a Complex World*. Burr Ridge, IL: McGraw-Hill Higher Education, 2000.

[Sullivan 99] Sullivan, K.; Knight, K.; Du, X.; & Geist, S. *Information Survivability Control Systems*, Proceedings of 21st International Conference on Software Engineering. Los Alamitos, CA: IEEE Computer Society Press, 1999.

[Thayer 90] Thayer, R. & Dorfman, M., eds. *System and Software Engineering*. Los Alamitos, CA: IEEE Computer Society Press, 1990.

[Trammell 95] Trammell, C. *Quantifying the Reliability of Software: Statistical Testing Based on a Usage Model*, 208-218. Proceedings of the Second IEEE International Symposium on Software Engineering Standards. Montreal, Quebec, Canada, August 21-25, 1995. Los Alamitos, Calif.: IEEE Computer Society Press, 1995.

[Tullock 03] Tulloch, Mitch. *Microsoft Encyclopedia of Security*. Redmond, WA: Microsoft, 2003.

[van der Meulen 00] van der Meulen, Meine. *Definitions for Hardware and Software Safety Engineers*. London, England: Springer, 2000.

[Vatis 01] Vatis, M. A. *Cyber Attacks During the War on Terrorism: A Predictive Analysis*. Hanover, NH: Institute for Security Technology Studies at Dartmouth College, 2001. http://www.ists.dartmouth.edu/ISTS/counterterrorism/cyber_attacks.htm

[Walton 02] Walton, G.; Hevner, A.; Linger, R.; & Pleszkoch, M. *Computational Quality Attributes for Distributed System Operation*, A publicar.

[Weidenhaupt 98] Weidenhaupt, K.; Pohl, K.; Jarke, M.; & Haumer, P. *Scenarios in System Development: Current Practice*, IEEE Software 15, 2 (March/April 1998): 34-45.

[Wolstenholme 90] Wolstenholme, E. F. *System Enquiry: A System Dynamics Approach*. ew York: John Wiley and Sons, 1990.

[Wolstenholme 93] Wolstenholme, E. F.; Henderson, S.; & Gavine, A. *The Evaluation of Management Information Systems: A Dynamic and Holistic Approach*. New York: John Wiley and Sons, 1993.

[Yacoub 99] Yacoub, S.; Cukic, B.; & Ammar, J. *Scenario-Based Reliability Analysis of Component-Based Software*, Proceedings of the 18th IEEE Symposium on Reliable Distributed Systems. Switzerland, October 1999. Los Alamitos, CA: IEEE Computer Society Press, 1999.

[Young 87] Young, W.D. & J. McHugh. *Coding for a Believable Specification to Implementation Mapping*, 140-149. Proceedings of the 1987 IEEE Symposium on Security and Privacy. Oakland, CA, Apr. 27-29, 1987. Los Alamitos, CA: IEEE Computer Society Press, 1987.