Desarrollo de un enfoque de trabajo para el Análisis y Diseño de Sistemas Discretos y Dinámicos

Aplicación a la Simulación de la demanda eléctrica de la ciudad de Salta

ALUMNO:
Lic. Franco Zanek

DIRECTOR:
Dra. Judith Franco

CO-DIRECTORES:
Esp. Marcia Mac Gaul
Dr. Gustavo Rossi

Tesis para acceder al título de Magister en Ingeniería de Software

Universidad Nacional de La Plata – Facultad de Informática

Julio - 2019

INDICE

Agradecimientos

A Dios, por darme la vida y darme el camino que hasta hoy he recorrido.

A mi familia, por todo el cariño, amor, educación, apoyo y ejemplo que me han brindado, con lo cual he logrado todas las satisfacciones por las cuales hoy, me siento orgulloso como persona.

A mis directores la Dra. Judith Franco, Esp. Marcia Mac Gaul y Dr. Gustavo Rossi, por éste trabajo de tesis, por ser excelentes personas y haberme brindado su apoyo en todo momento.

A CAMMESA, EDESa y a la Secretaría de Energía de la Provincia de Salta, por brindarme los datos y herramientas necesarias para poder llevar a cabo este trabajo.

A todas aquellas personas, que de una u otra manera estuvieron allí para ayudarme durante el proceso de elaboración de la tesis.

Resumen

La reunión de las disciplinas orientadas a los problemas y las soluciones puede llevar a importantes avances en ambas áreas. Las redes de Petri (RdP) proporcionan un medio excelente para modelar aspectos concurrentes y se han extendido de muchas maneras para hacer frente a muchos problemas. Las RdP se han aplicado con éxito muchas veces a varios problemas de ingeniería de software. Sin embargo, las dos disciplinas no pasan por un período de fertilización cruzada particularmente fuerte. Este trabajo trata de analizar algunos aspectos de la ingeniería de software, señalando aspectos en los que las RdP se han propuesto o se pueden proponer como soluciones a problemas críticos.

En esta tesis, se propone el desarrollo de un enfoque de trabajo para realizar el análisis y diseño de sistemas discretos, dinámicos y estocásticos. Estos sistemas, se caracterizan por estar íntimamente relacionados con restricciones temporales y concurrentes, que por las características de los modelos desarrollados por UML, no pueden ser representadas; con lo cual es necesario complementar las herramientas con otras, que permitan modelar las características antes mencionadas; una de estas, son las RdP.

Una RdP es un lenguaje útil para analizar y modelar formalmente varios sistemas. Recientemente, muchas RdP dedican sus esfuerzos a mejorar y extender el poder expresivo de las RdP. Uno de estos esfuerzos es extender las RdP con conceptos orientados a objetos. Un paradigma orientado a objetos proporciona conceptos excelentes para modelar problemas del mundo real. Los conceptos orientados a objetos nos permiten construir sistemas de software de forma fácil, intuitiva y natural. Se sugieren varias RdP de alto nivel con el concepto de objetos. Estas redes no son totalmente compatibles con el concepto orientado a objetos, por lo que no pueden llamarse RdP orientadas a objetos.

La sintaxis formal y la semántica del enfoque propuesto se explican en detalle, adoptando una amplia gama de características del análisis y diseño orientados a objetos. Además, este enfoque es compatible con una variedad de mecanismos de análisis, como los métodos de descomposición, red e incrementales de los sistemas en evolución, el despliegue, a un nivel más bajo de la RdP y el análisis de accesibilidad incremental para los modelos desarrollado.

Por último, se demuestra la eficiencia y la utilidad del enfoque desarrollado, a partir de la aplicación del mismo al caso de estudio, esto es, la simulación que explica el comportamiento y la demanda eléctrica residencial de la Ciudad de Salta, a partir de la cantidad y tipo de artefactos presentes en cada vivienda y el comportamiento humano para el encendido y apagado de los mismos.

Capítulo 1: Introducción

1.1 Motivación

La complejidad del proceso de ingeniería del software ha presentado a los desarrolladores e investigadores un desafío significativo, tornándolo a este conjunto de tareas algo complejo de llevar a cabo. Esto se debe a diversos factores, tales como la variedad de dominios de aplicación, de plataformas y de métodos y herramientas que soportan al proceso de desarrollo, entre otros aspectos.

De particular importancia en el ciclo de vida de un proyecto de software es la necesidad de facilitar una transición suave entre las distintas fases del proceso de desarrollo; la dicotomía entre las etapas de diseño, análisis e implementación se manifiesta en la multiplicidad de formalismos, lenguajes y herramientas que se requieren para cada una de ellas, independientemente de la metodología con la que se esté trabajando. El limitado alcance de estas herramientas y su bajo acoplamiento con dominios específicos es una de las principales fuentes de heterogeneidad entre los modelos creados en la fase de diseño y los modelos requeridos para el análisis.

Una de las principales preocupaciones de los desarrolladores es cómo facilitar la interoperabilidad entre los modelos que pertenecen al diseño y los requeridos por la fase de análisis; otro requisito es cómo integrar las herramientas de software correspondientes. En este sentido, el modelo de interoperabilidad y la integración de herramientas se consideran factores críticos para reducir la complejidad en el desarrollo de software.

Desde que se comenzó a trabajar en el desarrollo de un sistema software y hasta la actualidad, se han introducido varios lenguajes y formalismos para apoyar el desarrollo de sistemas en general y el diseño y análisis de los mismos en particular. El diseño, ha sido facilitado por la introducción de Lenguaje Unificado de Modelado (UML, por sus siglas en inglés); entre una de los lenguajes más usuales. Sus construcciones han conferido a UML un papel privilegiado en el diseño de sistemas en una variedad de dominios, incluyendo redes, modelado de negocios y seguridad. La elección de UML para el diseño de software también es facilitada por las herramientas de software UML ampliamente disponibles como ArgoUML y Poseidón. Una deficiencia de UML, sin embargo, es su falta de apoyo para el análisis formal; esta característica ha llevado a los desarrolladores de software a confiar y apoyarse más en otros lenguajes y herramientas, para suplir esa falencia.

En la actualidad resulta preocupante comprobar que, en muchos casos, son los intereses comerciales los que orientan el mercado aún en direcciones que se contraponen con lo que indicarían otros puntos de vista, incluido el técnico. Se

acentúa una visión simplista e ingenua de que el desarrollo de sistemas complejos se simplifica con la sola utilización de los métodos que han dado buenos resultados en otros campos de aplicación, o de que todo se resuelve incrementando los recursos de hardware. Coexiste, felizmente, otra realidad determinada por el ya mencionado amplio espectro de buenas técnicas y herramientas de análisis y diseño de sistemas que han sido desarrolladas y están disponibles. El esfuerzo debe ahora orientarse a hacer que trasciendan los ámbitos de los centros de investigación y se incorporen al mercado cotidiano.

Existe una creciente demanda por métodos que brinden mayor formalismo al desarrollo de un sistema, que no solo resuelvan el problema de cómo manejar la fusión entre el software y el hardware, sino que también permitan manejar los requerimientos de estos tipos de sistemas. Esta demanda, se ve en aumento especialmente en los sistemas relacionados con eventos estocásticos, sistemas automáticos y otros similares. Para poder satisfacer esta demanda, empezaron a divulgarse nuevas técnicas, herramientas y métodos, algunas de los cuales están ligados con alguna técnica de desarrollo de software o bien conectados con un área específica de la Ingeniería de Software.

Los sistemas discretos, dinámicos y estocásticos son sistemas que se caracterizan porque el estudio de los cambios de estados, se producen en instantes periódicos de tiempo, que se producen por la ocurrencia de algún evento aleatorio. En la actualidad, estos tipos de sistemas se encuentran en crecimiento y abarcan una gran variedad de dominios. Estos tipos de sistemas se pueden observan en: cadenas de producción, sistemas logísticos, transporte terrestre y aéreo, redes de comunicaciones, entre otros.

El modelado de estos tipos de sistemas, es un fenómeno relativamente reciente y en auge, porque permite abordar con éxito un proceso de mejora continua. Requieren formalismos que sean capaces de captar características como: concurrencia, paralelismo, operaciones asincrónicas, conflictos y compartimiento de recursos. Además, estos sistemas se caracterizan porque los requerimientos se encuentran totalmente claros y completos a la hora de inicio del proceso de desarrollo. De esta forma, se han incorporado a la creciente demanda de software especializado que se verifica en la actualidad. Además, el abaratamiento del hardware y el incremento de la densidad de integración, hace propicia la realización de sistemas cada vez más complejos y es de esperarse que esta tendencia se acentúe en el futuro próximo.

Así, la necesidad de dar respuesta a esta demanda no ha hecho más que incorporar nuevas exigencias al gran esfuerzo que se viene realizando para superar definitivamente la llamada "*crisis del software*" y que ha justificado una muy importante inversión de recursos, tanto intelectuales como económicos. Estos esfuerzos han estado orientados a establecer nuevas técnicas, métodos y herramientas que permitan tanto mejorar la calidad de los desarrollos de

software como así también a aumentar la productividad de los procesos conducentes a tales desarrollos, aumentando eficiencia y eficacia.

Como se mencionó anteriormente, UML es un lenguaje de modelado muy utilizado actualmente en ingeniería de software, ya que se caracteriza por tener una notación sencilla y con una fuerte base en la gráfica, consolida varios conceptos relacionados con el desarrollo de cualquier sistema software. Sin embargo, al intentar un bosquejo inicial de los requerimientos identificados, hasta llegar a los diagramas UML, resulta obvio que necesitamos ser capaces de modelar aspectos dinámicos y estáticos de un proceso. El tipo de aspectos estáticos que se necesitan expresar están cubiertos en UML sólo por diagramas de clase. A la hora de considerar el aspecto dinámico, se podría pensar en modelar esto tipos de sistemas mediante diagramas de actividades, secuencia o de estado; si bien estos diagramas ofrecen una sintaxis adecuada para modelar las entradas y salidas de sistemas estáticos, no proveen las restricciones relacionadas con el tiempo, aspecto fundamental en cualquier sistema dinámico. Tampoco proveen un tratamiento para los elementos relacionados con los aspectos probabilísticos.

A pesar de esto, UML es muy flexible y adaptable debido a su mecanismo de extensión. Es por esto que, para involucrar estas propiedades, muchos mecanismos han sido propuestos y desarrollados para el modelar estos sistemas. Dentro de estos se encuentran las RdP. Estas permiten, mediante herramientas gráficas y con un soporte matemático, representar modelos de sistemas que poseen las propiedades descriptas.

Las RdP fueron introducidas en la literatura, a partir de la tesis doctoral "*Comunicación con autómatas*", del alemán Carl Adam Petri en 1962, en la que se establecen los fundamentos para el desarrollo teórico de los conceptos básicos de las RdP, como una herramienta para simular las propiedades dinámicas de sistemas complejos, mediante modelos gráficos de procesos concurrentes. Las RdP son consideradas como una herramienta formal para dar soporte a todo el ciclo de vida del desarrollo de sistemas discretos, dinámicos y estocásticos. Se puede observar su uso en sistemas de manufacturación, arquitecturas multiprocesador, sistemas de comunicación y también en la implementación de confiables y eficientes programas concurrentes. Además, las RdP son lo suficientemente flexibles para permitir cambios en la red o bien sugerir un cambio en los casos de uso, en caso de detectar algún error en la especificación de los requerimientos.

1.2 Objetivos

La contribución de esta tesis es:

"*El desarrollo de un enfoque de trabajo para el análisis y diseño de sistemas discretos, dinámicos y estocásticos*"

Para lograr este objetivo, se define formalmente un enfoque de trabajo de RdP combinado con Paradigma Orientado a Objetos, para realizar el análisis y diseño de sistemas dinámicos, estocásticos y discretos. Con el desarrollo de este enfoque se logran optimizar las tareas vinculadas con el desarrollo de los sistemas antes mencionados, logrando un entorno de trabajo que facilite los desarrollos y permita avanzar con mejoras, relativas a las etapas de implementación y prueba. Además, este trabajo se caracteriza por brindar una transparencia, facilidad de comprensión del sistema, a través de los modelos desarrollados, lo que facilitará una futura incorporación que pueda llegar a sufrir el sistema, dada la flexibilidad del enfoque al proceso de desarrollo.

En este trabajo, se presenta un caso práctico, donde se aplican los conceptos al modelado de un sistema que permita simular el consumo de energía eléctrica, para el sector residencial, de la Ciudad de Salta, Argentina.

Para cumplir con el objetivo general, se plantean los siguientes objetivos específicos:

- Estudiar el alcance de UML y RdP como herramientas independientes para el análisis y diseño de sistemas discretos y dinámicos.
- Elaborar un enfoque de trabajo para el análisis y diseño de sistemas discretos y dinámicos, soportado en el uso de las RdP como herramienta complementaria a UML.
- Aplicar ambas herramientas a un caso práctico de modelado de un sistema estocástico, discreto y dinámico.

1.3 Organización de la tesis

En el capítulo 2 se presenta el marco teórico que abarca los conceptos de RdP y su integración dentro de la ingeniería de software. Además, se presenta un análisis de la literatura referente a la aplicación de las RdP en el desarrollo de sistemas software, y su vinculación con UML. Posteriormente, en el capítulo 3 se procede a realizar una descripción detallada del enfoque propuesto, brindando la sintaxis necesaria para poder aplicarlo exitosamente. Seguidamente, en el capítulo 4, se realiza una descripción del sistema bajo estudio y se muestra la aplicación del enfoque propuesto a dicho sistema. Para posteriormente, en el capítulo 5, realizar no solo un análisis estadístico de los resultados obtenidos por la simulación, sino también un análisis de los resultados obtenidos a partir de la aplicación del enfoque desarrollado. Finalmente, se presenta una conclusión del trabajo, realizando un cierre del mismo; conjuntamente con la bibliografía referenciada.

Capítulo 2: Marco Teórico

Esta sección presenta todos los aspectos teóricos necesarios para abordar con éxito la lectura de esta tesis, como así también se presenta una revisión de los trabajos presentados por diversos autores, en el tópico del presente trabajo.

2.1 Introducción

La ingeniería de software no se centra en una técnica o modelo particular para brindar una solución a un problema determinado, sino que selecciona la mejor solución que se adapte a los requerimientos en cada problema y contexto; por esto la solución de un problema puede llegar a ser un sub-óptimo para otro problema o bien no ser solución de ningún otro problema. Un ejemplo claro donde se aplica esta situación, son la especificación y diseño de software, ya que en los años ochenta se encontraban en auge el análisis estructurado, que en los noventas fue sustituido por los paradigmas orientados a objetos y actualmente se sustituyeron por las diversas propuestas de metodologías ágiles. Por otro lado, por ejemplo, los sistemas basados en cliente-servidor que son beneficiosos para la resolución de un tipo de problemas sin embargo para otro grupo de problemas.

Eso se debe a que este tipo de ingeniería no se centra en un modelo particular, o una técnica para brindar una solución de todos los problemas, sino que selecciona las mejores soluciones que se adaptan a los requerimientos y contextos de cada problema. Se podría decir que estas disciplinas se centran en los problemas más que en las soluciones de los mismos.

Por otro lado, las investigaciones basadas en las RdP hacen foco en una solución. Estas investigaciones, no están dirigidas por un problema, donde lo que se busca es determinar soluciones al mismo; sino que brinda los conceptos teóricos necesarios, para poder resolver problemas de diferente naturaleza. Estas, se centran en el estudio de diversas posibilidades presentadas en la teoría y propone un marco de trabajo que brinda soluciones a diferentes dominios. Los resultados de estos trabajos, demuestran que las RdP pueden ser utilizadas para brindar respuestas en el campo de la ciencia de la computación, química, biología, diseño de hardware, especificación de requerimientos de software, computación distribuida, entre otros. Las RdP, pueden ser clasificadas como una disciplina que se centra en la solución, que se caracteriza por ser homogénea y posee un marco conceptual bien definido y un conjunto estable de herramientas [1].

Ahora, si nos centramos en la combinación de las disciplinas basadas en soluciones con aquellas centradas en los problemas, puede generar grandes beneficios para ambos campos: los primeros pueden encontrar las soluciones más eficientes para los problemas, mientras que, por otro lado, las disciplinas centradas en soluciones pueden producir un nuevo estímulo en el campo. A partir del análisis de la bibliografía, podemos observar la importancia del análisis

de requisitos y modelado de sistemas. En el proceso del desarrollo de software ha resultado en un aumento gradual del uso de métodos formales como las RdP.

La ingeniería de software y las RdP confluyen en varios ámbitos con el objetivo de brindar una solución mejor a los problemas planteado. Dentro de la primera, las RdP fueron utilizadas para la especificación de requerimientos, ingeniería reversa, diseño de interfaces de usuarios, modelado y análisis de sistemas críticos, sistemas distribuidos y sistemas de tiempo real, administración de procesos de software y evaluación de la performance de sistemas software [1].

2.2 Redes de Petri

Las RdP surgen de la tesis doctoral "Kommunikation mit Automaten" del alemán Carl Adam Petri presentada en la facultad de matemáticas y física de la Universidad Técnica de Darmstadt en 1962.

Una RdP (Petri Net, PN por sus siglas en inglés) es una herramienta gráfica y matemática que ayuda a describir relaciones entre condiciones y eventos presentes en los sistemas del mundo real. Las características de un sistema que son modeladas utilizando RdP son algunas de las siguientes: sincronización de procesos, eventos asíncronos, operaciones secuenciales, operaciones concurrentes, conflictos con recursos compartidos, procesos distribuidos, procesos paralelos, procesos no determinísticos y/o estocásticos. Además, constituyen una potente herramienta de modelado, descripción y análisis, tanto gráfico como matemático aplicable a una gran diversidad de sistemas. Como herramienta gráfica, las RP sirven de soporte para la comunicación visual, al igual que los diagramas de flujo de datos y diagramas de bloques funcionales abordados en el capítulo anterior, además, los ¨*tokens*¨ o marcas que se utilizan en estas redes sirven para simular las actividades dinámicas y concurrentes.

Como herramienta matemática, permiten modelar y analizar ecuaciones de estados, ecuaciones algebraicas y otros modelos matemáticos. También facilita el análisis y la verificación de un gran número de propiedades presentes en los sistemas. El análisis de estas propiedades permite la verificación de requisitos funcionales de los modelos, posibilitando la detección de posibles errores y por tanto la corrección de estos antes de pasar a la etapa de implementación. La utilización de esta herramienta de modelado se hace efectiva en áreas como:

- Fabricación Flexible y Sistemas de Control Industrial.
- Sistemas de eventos discretos.
- Modelado y análisis de sistemas de ¨*hardware*¨, ¨*software*¨ y bases de datos distribuidos.
- Sistemas de tolerancia a fallos.
- Redes Neuronales Artificiales.
- Sistemas de memoria multiprocesador y flujo computarizado de datos.
- Programas concurrentes y paralelos.

Un sistema modelado a través de RdP puede ser analizado de manera formal, obteniendo información de su comportamiento dinámico. Además, es posible verificar propiedades y condiciones. Los eventos son acciones que se presentan en el sistema y lo llevan a un estado. Es posible describir un estado como un conjunto de condiciones; para que cierto evento ocurra es necesario que ciertas condiciones se cumplan; a éstas se les llama precondiciones del evento, así mismo, la ocurrencia del evento puede llevar a otras condiciones; éstas se conocen como postcondiciones.

Al modelar con RdP es necesario identificar las condiciones (pre y postcondiciones), así como los eventos que pueden suceder en el sistema. De esta manera se traslada el sistema de la vida real al modelo en RdP. La tarea de modelado es una actividad subjetiva. Es decir, el modelado de un sistema puede realizarse de diferentes maneras. Así cuando diferentes personas realizan el modelo de un mismo sistema, los modelos resultantes pueden diferir considerablemente; sin embargo, ambos estarán modelando el comportamiento del sistema. Al tener las condiciones necesarias para que ciertos eventos del sistema sucedan, es posible modelar de forma modular, y de esta manera relacionar los modelos con otras condiciones; para esto es necesario conocer la estructura de la red.

Informalmente, una RdP es un grafo dirigido, bipartito que incluye una serie de lugares, un conjunto de transiciones y un conjunto de arcos dirigidos. Es decir, que en el grafo solo hay dos clases de vértices llamados lugares y transiciones. Los lugares pueden contener uno o más elementos denominados *tokens*. El marcado de una RdP asigna a cada lugar un entero no negativo, que coincide con el número de elementos que contendrá cada lugar. En tal caso, decimos que se trata de una RdP marcada.

Al modelar una situación, los lugares representan condiciones o el estado en que se encuentra el sistema modelado en un momento determinado. Las transiciones representan eventos, y la presencia de al menos un elemento en un lugar (condición), indica que tal condición se cumple.

A partir de la lectura de los materiales bibliográficos del tema, podemos encontrar que la definición clásica de las RdP ordinarias, está formada de cinco componentes: Un conjunto de lugares o estados P, un conjunto de transiciones T, la función de entrada I, la función de salida O y el número de marcaciones (o *tokens*) de cada lugar de la red. Las funciones de entrada y salida relacionan las transiciones y los lugares. La definición, expresa en términos matemáticos, queda expresada de la siguiente manera:

RP= (P, T, I, O, Mo)

Donde:

P= $\{p_1, p_2, \dots, p_n\}$: conjunto finito de lugares o estados (representados por circunferencias).

T= $\{t_1, t_2, \dots, t_m\}$ es un conjunto finito y no vacío de transiciones (representados generalmente por barras o rectángulos).

I: es una Función de incidencia previa la cual representa los pesos de los arcos de entrada a las transiciones. $I: P\ x\ T \rightarrow N_1$

O: es una Función de incidencia posterior la cual representa el peso de los arcos de salida de las transiciones. $O: T\ x\ P \rightarrow N_2$

Mo: representa la marcación inicial Mo: $P \rightarrow N_3$

$(P \cap T) = \emptyset$

$(P \cup T) \neq \emptyset$

$N_1, N_2 \in \mathbb{N} \geq 1, N_3 \in \mathbb{Z}_0^+$

Como ejemplo, podemos definir la siguiente RdP básica:

Definición de lugares y transiciones

$P = \{p_1, p_2, p_3, p_4\}$

$T = \{t_1, t_2, t_3\}$

Definición de la función de incidencia de entrada a las transiciones

$I(p_1, t_1) = 2\ , I(p_i, t_1) = 0\ i = 2,3,4$

$I(p_2, t_2) = 1\ , I(p_i, t_2) = 0\ i = 1,3,4$

$I(p_3, t_3) = 1\ , I(p_i, t_3) = 0\ i = 1,2,4$

Definición de la función de incidencia de salida de las transiciones

$O(t_1, p_2) = 2\ , O\ (t_1, p_3) = 1, O(t_1, p_i) = 0\ i = 1,4$

$O(t_2, p_4) = 1\ , O(t_2, p_i) = 0\ i = 1,2,3$

$O(t_3, p_4) = 1\ , O(t_3, p_i) = 0\ i = 1,2,3$

Marcación inicial de los lugares de la red

$Mo = (2,0,0,0)$

2.2.1 Representación gráfica

La representación gráfica de una RdP es importante ya que si se observa el modelo del sistema en forma gráfica y la manera en que cambia de un estado a otro, es posible mantener la atención y obtener una perspectiva más clara del análisis del problema. En la figura 1 se muestra la representación gráfica de los elementos que forman una RdP.

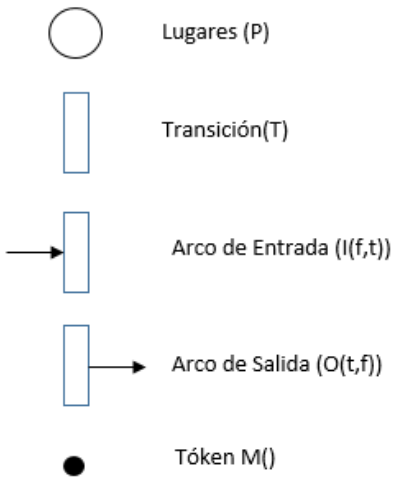


Figura 1 Representación gráfica de una RdP

Los lugares son representados con círculos, las transiciones se representan mediante una barra o un rectángulo, los arcos dirigidos que conectan a los lugares con las transiciones y viceversa son representados con flechas. Los *tokens* son representados con pequeños círculos de color negro que van dentro de los lugares.

Un arco dirigido desde un lugar p_j (transición t_i) a una transición t_i (lugar p_j) define a p_j como un lugar de entrada (salida) de t_i, el cual describe como $w(p_j, t_i) = w(t_i, p_j) = 1$. Si $w(p_j, t_i) = k$ (o $w(t_i, p_j) = k$), entonces existen k arcos dirigidos (paralelos) que conectan el lugar p_j con la transición t_i (o que conectan la transición t_i con el lugar p_j). Generalmente en un esquema gráfico, los arcos paralelos que conectan a un lugar (transición) con una transición (lugar) se representan por un solo arco dirigido, etiquetado con el número de arcos que representa a su peso k. Un lugar que contiene un punto en su interior representa a un lugar que contiene un *token*. La capacidad de almacenamiento de *tokens* en cada lugar es infinita, y para denotar el número de *tokens* o marcado de un lugar se utiliza M(p).

Volviendo al ejemplo descripto anteriormente, la representación gráfica del mismo queda como:

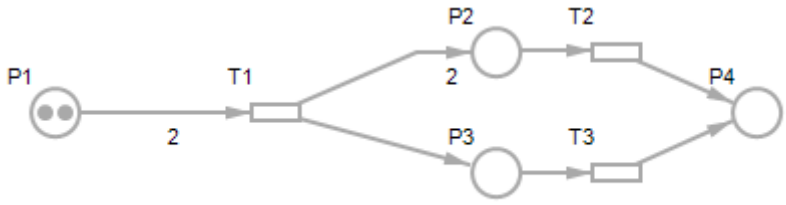


Figura 2 Representación gráfica del ejemplo de la RdP descripto en la sección 2.2

2.2.2 Regla de Disparo

La ejecución de una RdP está controlada por el número y la distribución de *tokens* en la red. Al cambiar la distribución de *tokens* en los distintos lugares, se puede estudiar el comportamiento dinámico del sistema modelado; este cambio de *tokens* puede reflejar la ocurrencia de eventos o la ejecución de operaciones.

La ejecución de la RdP se activa disparando sus transiciones. Cuando una transición es disparada se remueven *tokens* de los lugares de entrada y se crean *tokens* en los lugares de salida. El número de *tokens* que son removidos es igual al peso del arco que conecta al lugar de entrada con la transición que fue disparada. De igual manera el número de *tokens* creados está dado por el peso del arco que conecta la transición con el lugar de salida. A partir de las reglas que se definen a continuación, se determina la activación y habilitación de una transición:

- ***Regla de habilitación:*** se dice que una transición t está habilitada si cada lugar de entrada p de t contiene al menos el número de *tokens* igual al peso del arco dirigido que conecta p con t, es decir, $M\ (p) \geq\ I\ (p,t)\ \forall\ p\ \subset P$. Si $I(p,t) = 0$ entonces t y p no están conectados, por lo que no nos importa el marcado de p al considerar el disparo de t.

- ***Regla de disparo:*** solo las transiciones habilitadas pueden disparar. El disparo de una transición habilitada t elimina de cada lugar de entrada p_1 el número de *tokens* igual a $I(p_1,t)$ y deposita en cada lugar de salida p_2 el número de *tokens* igual a $O(t,p_2)$.

Desde el punto de vista matemático, disparar una transición produce una nueva marcación en los lugares de salida.

Ecuación 1 Modificación de las marcaciones de un lugar

$$M'(p) = M(p) - I(p,t) + O(t,p)\ \ \forall\ p\ \in P$$

En la figura 3 se muestra la regla de disparo y algunos ejemplos. Cuando se omite el peso del arco por convención el peso es 1, no sucede así con el número de *tokens*. Es decir, si no existen *tokens* el número de *tokens* es 0. Es así como en el ejemplo mostrado en la figura 3 a) la red no se habilitará nunca ya que el peso del arco es 1 y no es ni mayor ni igual a 0, que es el número de *tokens* en el lugar. Un caso similar se da en el ejemplo de la figura 3 b). Sin embargo, la figura 3 c) muestra un ejemplo en el que la transición sí se activará ya que el número de *tokens* en el lugar es mayor al peso de su transición. Una transición activada puede o no dispararse, dependiendo de la interpretación adicional que tenga la transición.

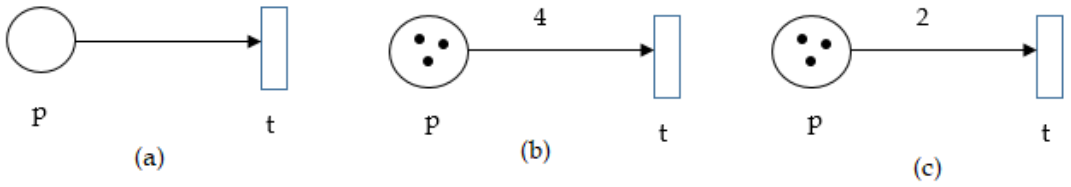


Figura 3 Regla de disparo

En la figura 4 se muestra la ejecución de una RdP sencilla. En ella podemos verificar cómo el peso de los arcos decide el número de *tokens* que son removidos del lugar de entrada y el número de *tokens* creados en el lugar de salida. También podemos observar que llega un momento en que el lugar de salida queda sin *tokens*, esto es después del tercer disparo. Esto provoca la deshabilitación de la transición y la ausencia de posteriores disparos en el sistema, hecho que podríamos interpretar como la finalización de nuestro sistema, o el agotamiento de sus recursos, esto es dependiendo de la interpretación con que modelamos nuestro sistema.

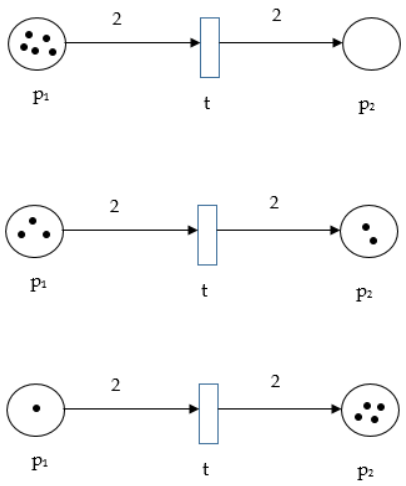


Figura 4 Ejemplo de ejecución de una RdP

2.2.3 Propiedades

En el análisis y diseño de modelos basados en RdP se tienen en cuenta un conjunto de propiedades [2], pudiendo clasificarse estas como:

- Propiedades dependientes del marcaje, o **propiedades funcionales**: Son las propiedades que dependen del marcaje inicial y reflejan el comportamiento dinámico del sistema.
- **Propiedades estructurales**: Son independientes del marcaje inicial de la red.

Entre las **propiedades estructurales** que más se analizan se encuentran **Vivacidad Estructural** y **Limitación Estructural** [2]**.** La primera asegura que

todas las transiciones puedan ser disparadas a partir de un marcaje inicial, la segunda asegura que el número total de ¨*tokens*¨ en la red sea finito y que no haya acumulación en ningún nodo. Otras propiedades son la capacidad de **Repetitividad**, la **Controlabilidad** y las **Invariantes S** y **T**, siendo estas últimas unas de las más importantes. En el caso de las Invariantes **S** descubren la estructura mínima de los circuitos que pueden compartir marcas independientemente de la evolución dinámica que experimente la red. Por su parte, las invariantes **T** definen el número de veces que cada transición debe dispararse para ir desde una marca **Mo** hasta esa misma marca inicial completando un ciclo entero.

Entre las **propiedades funcionales** más importantes se encuentran:

2.2.3.1 Alcance

En general, la pregunta es si el sistema modelado con una RdP exhibe todas las propiedades deseables según lo especificado en los requisitos. Para saber si el sistema modelado puede alcanzar un estado específico como resultado de un comportamiento requerido, es necesario encontrar una secuencia de activación de transición que vayan de la marca inicial M_0 a la marca deseada M_j, donde M_j representa el estado específico, y la secuencia de disparo representa el comportamiento funcional requerido.

En general, se dice que una marca M_j es accesible desde una marca M_i si existe una secuencia de disparos de transición que transforma M_i a M_j. Se dice que una marca M_j es inmediatamente accesible desde $M_i$si el disparo de una transición habilitada en Mi resulta en M_j. El conjunto de todas las marcas accesibles desde el marcado M se denota con $R(M)$.

2.2.3.2 Seguridad

En una RdP, los lugares a menudo se utilizan para representar áreas de almacenamiento de información en sistemas de comunicación o informáticos. Entonces, es importante poder determinar si las estrategias de control propuestas previenen los desbordamientos de estas áreas de almacenamiento.

La propiedad de las RdP, que ayuda a identificar la existencia de desbordamientos en el sistema modelado, es el concepto de delimitación. Se dice que un lugar p está limitado por $\boldsymbol{k}$ si el número de *tokens* en $\boldsymbol{p}$ es siempre menor o igual que $\boldsymbol{k}$ ($\boldsymbol{k}$ es un número entero no negativo) para cada marca M accesible desde la marca inicial M_0, es decir, M ∈ $R(M_0)$.

2.2.3.3 Vitalidad

El concepto de vitalidad está estrechamente relacionado con la situación de deadlock, que se ha situado ampliamente en el contexto de los sistemas

operativos informáticos. Una RdP que modela un sistema sin puntos muertos debe estar activa. Esto implica que para cualquier marca M alcanzable, cualquier transición en la red se puede disparar progresando a través de alguna secuencia de disparo.

Este requisito, sin embargo, podría ser demasiado estricto para representar algunos sistemas reales o escenarios que exhiben un comportamiento sin trabas. Por ejemplo, la inicialización de un sistema se puede modelar mediante una transición (o un conjunto de transiciones) que se dispara un número finito de veces. Después de la inicialización, el sistema puede mostrar un comportamiento libre de interbloqueo, aunque la RdP que representa este sistema ya no está activa como se especificó anteriormente. Por esta razón, se definen diferentes niveles de vida. Indica con $L(M_0)$ el conjunto de todas las secuencias de disparo posibles a partir de M_0. Se dice que una transición t en una RdP es:

1. L_0-live (o dead) si no hay una secuencia de disparo en $L(M_0)$ en la que t puede disparar.

2. L_1-live (potencialmente disparable) si t puede dispararse al menos una vez en alguna secuencia de disparo en $L(M_0)$.

3. L_2-live si t puede dispararse al menos k veces en alguna secuencia de disparo en $L(M_0)$ dado cualquier entero k positivo.

4. L_3-live si t puede dispararse infinitamente a menudo en alguna secuencia de disparo en $L(M_0)$.

5. L_4--live (o live) si t es L1-live (potencialmente disparable) en cada marca en $R(M_0)$.

2.2.4 Tipos de Redes de Petri

A lo largo de la literatura se han propuesto diversos tipos de RdP, para diversos motivos, que se detallan a continuación:

2.2.4.1 Redes de Petri Coloreadas

Este tipo de redes fue introducido por Kurt Jensen en 1981, agregando a la definición de RdP, la característica de identidad de *token*. Además, cada lugar y cada transición tienen un conjunto de colores adjuntos. Una transición puede dispararse con respecto a cada uno de sus colores. Al activar una transición, los *tokens* se eliminan de los lugares de entrada y se agregan a los lugares de salida de la misma manera que en las RdP originales, excepto que se especifica una dependencia funcional entre el color de la activación de transición y los colores de los *tokens* involucrados. El color adjunto a un *token* puede cambiarse mediante

una activación de transición y, a menudo, representa un valor de datos complejo. Las RdP coloreadas conducen a modelos de red compactos utilizando el concepto de colores.

2.2.4.2 *Redes de Petri Temporalizadas*

La necesidad de incluir variables de tiempo en los modelos de los varios tiempos de sistemas dinámicos es evidente, puesto que estos sistemas son de naturaleza de tiempo real. En el mundo real, casi todo evento está relacionado con el tiempo. Cuando una RdP contiene una variable de tiempo, se convierte en una RdP temporalizada. Para poder definir este tipo de red son necesarios tres tipos de especificaciones la estructura topológica que se define de igual manera que una RdP clásica. El etiquetado para una RdP temporalizada consiste en la asignación de valores numéricos a una o más transiciones, lugares o arcos. Las reglas de disparo se definen de manera diferente dependiendo de la manera que la RdP se etiqueta con variables de tiempo. Las reglas de disparo definidas para una RdP temporalizada controlan el proceso de mover los *tokens* alrededor.

Las variaciones antedichas conducen a varios tipos diferentes de RdP temporalizadas. Entre ellas, se encuentran las RdP determinísticas y las RdP estocásticas en las cuales las variables de tiempo se asocian a transiciones.

2.2.4.3 Redes de Petri Estocásticas

Una RdP estocásticas es una RdP clásica con una interpretación temporal que reduce el no-determinismo inherente al modelo. La interpretación temporal debe incluir la asociación de una duración temporal a las actividades (transiciones) y también una política de resolución de conflictos. Con respecto a la duración de actividades, se suele asociar una variable aleatoria a cada transición, que modela su tiempo de servicio. Con respecto a la resolución de conflictos, se pueden asociar tasas de encadenamiento a cada subconjunto de transiciones en conflicto para elegir la transición que debe dispararse al presente conflicto.

Las tasas de encaminamiento pueden definirse explícitamente mediante una anotación asociada a cada transición inmediata (transición cuyo tiempo de servicio es nulo y sirve para modelar decisiones), o bien pueden calcularse para el caso de transiciones temporalizadas (que tienen un tiempo de servicio nulo) usando una política de competición (la primera transición que termina su actividad es la elegida para ser disparada).

2.2.4.4 Redes de Petri Estocásticas Generalizadas

Las RdP estocásticas generalizadas extienden a las anteriores. Los tiempos de servicio son variables aleatorias distribuidas exponencialmente. Se incluyen transiciones inmediatas que pueden usarse para modelar conflictos con unas

tasas de encaminamiento determinadas. Separando así la política de resolución de conflictos de la duración de las actividades.
Formalmente, este tipo de redes como una 9-upla:

$$(P,T,I,O,H,G,M_0,W,P_i)$$

Donde:

- P es un conjunto de lugares.
- T es el conjunto de transiciones.
- I, O y H son relaciones describiendo pre-condiciones, pos-condiciones y condiciones de inhibición, respectivamente.
- G es una función de habilitación que, dada una transición inmediata y un estado del modelo, determina si la transición está habilitada o no.
- M_0 es el conjunto de marcaciones iniciales.
- W asocia a cada transición temporalizada un número real no negativo, que representa la respectiva tasa o retraso exponencial, y a cada transición inmediata un número real no negativo representando su peso.
- P_i asocia a cada transición inmediata un número natural que representa su nivel de prioridad.

2.2.4.5 Redes de Petri Estocásticas Generalizadas Etiquetadas

Para facilitar el trabajo y el modelado, se presenta un complemento a la definición de RdP introducido en la sección anterior, las RdP Estocásticas Generalizadas Etiquetadas (LS). Estas se definen como una 3-upla:

$$(S,\psi,\lambda)$$

Donde:

- S es la RdP Estocástica Generalizada definida en la sección anterior.
- $\psi : T \rightarrow L^T \cup \tau$ una función de etiquetado que asigna a cada transición una etiqueta.
- $\lambda : T \rightarrow L^P \cup \tau$ una función de etiquetado que asigna a cada lugar una etiqueta.
- τ un conjunto de etiquetas.

2.2.5 Las Redes de Petri y los Sistemas de eventos discretos, dinámicos y estocásticos

Entre los modelos discretos, estocásticos y dinámicos, las RdP demostraron ser muy útiles tanto desde el punto de vista académico como desde el punto de vista de la industria. Sin embargo, existe actualmente en la literatura un esfuerzo por parte de la comunidad científica e industrial por ampliar los límites de las RdP y ampliar su poder de modelado. Las principales contribuciones han aparecido en las siguientes áreas relacionadas con los sistemas de eventos discretos:

- ***Control:*** este es el problema central en la teoría de sistemas y consiste en aplicar entradas de control adecuadas a una planta para garantizar que su comportamiento satisfaga una especificación determinada. Para las RdP, una solución típica es diseñar un supervisor que deshabilite el disparo de algunas transiciones como ley de retroalimentación del comportamiento o estado observado de la planta: el sistema controlado (planta y supervisor) también se denomina sistema de circuito cerrado.

- ***Estimación de marcado:*** el problema es el de determinar formas eficientes de reconstruir el estado o la secuencia de disparo de una red en función de la ocurrencia de eventos observados y / o de la observación de marcado parcial.

- ***Diagnóstico de fallas:*** dada una red que modela un sistema posiblemente afectado por una falla, bajo el supuesto de que la aparición de una falla (modelada por una transición no observable) no se puede observar directamente, el problema es detectar e identificar la falla del comportamiento observado.

- ***Deadlock***: un deadlock representa un estado anómalo a partir del cual no es posible una evolución posterior. Este es un problema que aparece en muchos problemas de automatización y se deben adoptar estrategias apropiadas para evitarlo.

- ***Identificación:*** este problema consiste en determinar un sistema de RdP a partir de ejemplos / contraejemplos de su lenguaje, o de la estructura de su gráfica de accesibilidad (o cobertura).

2.3 Redes de Petri y UML

Los requisitos funcionales de las aplicaciones de software son sumamente importantes, pero no son los únicos. Existen sistemas, como los de tiempo real en los que son especialmente importantes o críticas las especificaciones de performance.

El desarrollo de sistemas complejos requiere métodos adecuados para modelar y evaluar el comportamiento a posterior. La medición de cualquier sistema o dispositivo desarrollado en el marco de una disciplina de ingeniería debería ser claramente identificada como una etapa del ciclo de vida del producto. Siendo la ingeniería de software una disciplina relativamente joven, la importancia del uso de metodologías bien establecidas, métodos formales y lenguajes y herramientas, ya han sido detectados. Desafortunadamente, los objetivos de performance todavía no se incluyen en las primeras etapas del ciclo de vida del software.

Siendo los requisitos de performance críticos para el éxito de los sistemas de software de hoy en día, muchos productos de software finales fallan en alcanzar

esos requerimientos. La práctica usual consiste en, primero, hacer funcionar el producto, luego hacerlo funcionar bien y finalmente hacerlo funcionar rápido. Pero esta práctica es en muchos casos muy costosa, ya que arreglar problemas de performance puede obligar a modificar el diseño inicial.

Sin embargo, muchos investigadores defienden el principio de que la performance debería ser incluida en el proceso de diseño del software desde el comienzo. Este principio es uno de los principales objetivos del International Workshop on Software and Performance, un foro para investigadores interesados en la intersección de la ingeniería de software y la evaluación de la performance.

El campo de investigación cuyo objetivo es la construcción de software con performance predecible por medio de la especificación y el análisis del comportamiento cuantitativo desde las primeras fases de desarrollo de un sistema y a lo largo de todo su ciclo de vida, se ha denominado Software Performance Engineering (SPE). Más concretamente, en la SPE se desarrollan modelos de performance desde temprano en el ciclo de vida de software para estimar la performance.

Dado que los objetivos de performance no están incluidos en la práctica usual de los ingenieros de software, empieza a hacerse la necesaria la integración del modelado de performance con las metodologías y herramientas existentes de desarrollo de software. Los modelos de performance a menudo se describen en algún formalismo matemático que proporciones las capacidades requeridas de análisis y simulación. Los modelos de Markov, las Redes de Cola, las Álgebras de Procesos Estocásticos y las RdP Estocásticas son probablemente los paradigmas de modelado de performance mejor estudiados. El éxito de los mismos está ligado a la disponibilidad y existencia de herramientas para asistirlos.

La industria del software, al estar muy preocupada sobre los problemas que implican la calidad del software, ha participado activamente en conjnto con el consorcio Object Managment Group (OMG) y con otros grupos de investigación, en el desarrollo de un estándar para incorporar a las especificaciones del software, ciertos atributos no funcionales relacionados con la perfomance, tiempo, fiabildiad, planificación, entre otros aspectos relacionados. Ahora es posible especificar perfomance a partir del lenguaje de UML.

UML es un estándar de modelado ampliamente aceptado en la industria. Sin extensiones, UML no permite modelar y evaluar propiedades de perfomance, rendimiento, tolerancia a fallos, entre otros. Sin embargo, UML es muy flexible y adaptable debido a su mecanismo de extensión. Este mecanismo, permite la definición de perfiles que, para distintos dominios de aplicación, mapean aspectos de dicho dominio a los elementos del metamodelo de UML.

Se han propuesto varios trabajos para combinar UML y un formalismo de modelado de performance para analizar aspectos cuantitativos de los sistemas de software. Todos ellos comparten los mismos principios básicos:

- El comportamiento y la arquitectura del sistema se describe mediante una serie de diagramas de UML que constituyen el diseño del mismo.
- Un análisis cuantitativo se lleva a cabo, si el modelo formal lo permite.
- El modelo formal se analiza usando técnicas de análisis cuantitativas ya desarrolladas.

A los fines de analizar y evaluar cuantitativamente los diagramas UML extendidos, existen dos estrategias, la primera que consiste en desarrollar y aplicar un análisis que funcione directamente sobre el modelo UML. Mientras que la segunda estrategia, consiste en el mapeo de los modelos de UML en un modelo de performance previamente establecido.

Principalmente, se han utilizado tres paradigmas como medios para aportar formalismo a los modelados: RdP estocásticas, Álgebras de Procesos estocásticas y Redes de Colas. Actualmente, diversos investigadores vienen realizando contribuciones donde proponen diversos métodos y técnicas para la traducción o el mapeo de diagramas de comportamiento de UML en una RdP estocástica, ya que con su simulación es factible realizar la evaluación cuantitativa del modelo.

2.4 Estado del Arte

Para realizar esta revisión, fue necesario analizar los repositorios actuales, con el fin de determinar la literatura vigente sobre el tema y utilizarla de base para la realización de este trabajo. En ese proceso, se consideraron todos aquellos artículos que presentaban el desarrollo de simulaciones en diferentes áreas, utilizando RdP. Como así también aquellos artículos que presentan una aproximación a la integración de RdP con UML.

Además, para el proceso de selección de artículos se tuvo en cuenta el cálculo de la tasa de confiabilidad. Para esto, se decidió utilizar el método propuesto por Cohen´s Kappa [3], el cual estima que la tasa de confiabilidad mínima para considerar un trabajo aceptado, debe ser del orden de 0.7. En base a esto, se procedió a analizar los artículos descartando aquellos que tenían una tasa de confiabilidad inferior a lo estipulado anteriormente, lo cual se evidenciaba en el hecho que dichos artículos presentaban mayor cantidad de rechazos que aceptación. En cualquier otro caso, el artículo era considerado para un análisis posterior. Una vez realizado el proceso de selección se continuó con el estudio de los artículos que pasaron el filtro anterior, analizando el resumen y se continuaba con la lectura de aquellos que eran considerados adecuados, a partir del resumen expuesto.

En [4] los autores presentan la aplicación de las RdP como método de validación de requerimientos durante las etapas del desarrollo del Proceso Unificado, para el diseño de sistemas de información. En este trabajo, los autores proponen la captura de los casos de uso mediante modelos elementales de las RdP, permitiendo detectar anomalías en las redes y su correspondiente caso de uso. A partir de la ejecución de este trabajo, los autores concluyen que las herramientas desarrolladas proporcionan un gran avance en la validación de requerimientos, sin embargo, advierten en este trabajo que se encuentra en fases tempranas de desarrollo y planean desplegar una herramienta "add on" para la validación de requerimientos de sistemas pequeño o de mediana envergadura.

Siguiendo la misma línea de trabajo, Zhao J. y Duan Z. [5], un enfoque de trabajo, que tiene como objetivo superar las limitaciones de la redacción de los casos de uso mediante el desarrollo basado en modelos y la utilización de RdP controladas y cronometradas, con el fin de brindar una descripción formal y los mecanismos de verificación de los casos de usos, con el fin de determinar fallas o errores en los requerimientos. La metodología propuesta, consiste en lo siguientes:

1- Determinar el escenario del caso de uso.
2- Realizar la conversión de dicho escenario a una RdP.
3- Ejecutar la validación de la red, verificando lo siguientes aspectos:
    a. Completitud
    b. Consistencia
    c. Ambigüedad
4- Si no se encuentra errores, construir el modelo Independiente de Plataforma (PIM). Caso contrario volver a 1.

En la figura 5, se muestra un detalle de la metodología propuesta.

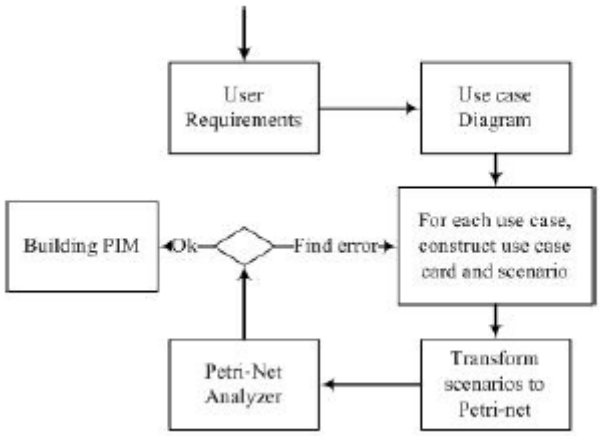


Figura 5 Descripción de la metodología propuesta en [5]

Por otro lado, en [6] los autores desarrollan un enfoque de trabajo para determinar la correspondencia entre los diagramas de secuencia y las RdP con tiempo, con el fin de validar los requerimientos en sistemas de tiempo real con restricciones de tiempo, con el fin de determinar potenciales problemas y así reducir la propagación de fallas desde la primera etapa de especificación hasta el código final. Para ello, los autores toman como punto de partida los diagramas de secuencia y realizan una descomposición en elementos básicos (línea de vida, mensajes y fragmentos combinados) para realizar una posterior equivalencia con las RdP, utilizando la notación MARTE. Luego de la aplicación del enfoque a un sistema de oxímetro de pulso[1], los autores concluyen que el enfoque desarrollado proporciona resultados prometedores para el modelado, análisis y verificación de sistemas de tiempo reales.

En la misma línea de trabajo, en [7], definen un conjunto de reglas de mapeo entre los diagramas de actividad y las RdP de tiempo coloreadas, con el fin de proporcionar herramientas de trabajo para modelizar y analizar sistemas complejos. En este artículo, los autores se centran en la gestión de los recursos, incluido en el tiempo y las propiedades no funcionales de los sistemas, como ser la robustez, confiabilidad, entre otras. Siguiendo la misma línea de encontrar una correspondencia y mapear los diagramas de UML y las RdP, están los trabajos de [8] donde los autores tienen como objetivo determinar las reglas de mapeo con las RdP coloreadas y determinan la eficacia de esta relación a partir de la aplicación del enfoque obtenido, al modelado de un sistema de gestión de la energía en las industrias de conserva.

En la misma línea de investigación se encuentran los trabajos propuestos por [9-11]; en este último se establece la relación entre los diagramas de actividad que provee UML y las RdP a través de unas reglas sencillas de asociación cada una de las actividades del diagrama corresponden a un lugar en la RdP asociada y cada una de las acciones del diagrama se reflejan como las transiciones en la red, en las figuras 6-8 se muestran ejemplos de estas correspondencias.

[1] Aparato médico encargado de medir indirectamente la saturación de oxígeno en la sangre, para paciente que poseen problemas cardiovasculares. [12]

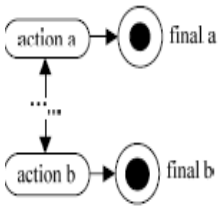
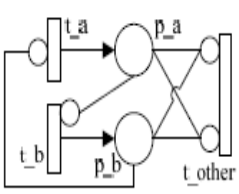
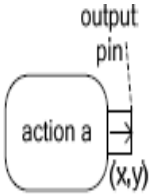


Figura 6 Regla de mapeo para nodos finales multiactividades [11]





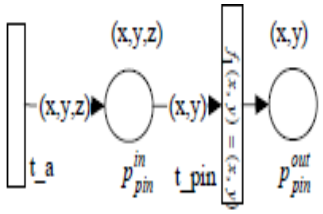


Figura 7 Reglas de mapeo para salidas [11]

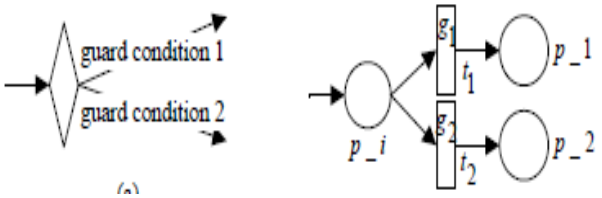


Figura 8 Reglas de Mapeo para situaciones condicionales [11]

En [13], los autores parten desde los diagramas de estado y utilizando las expresiones matemáticas provistas por las RdP, logran obtener un modelo equivalente y con mayor grado de formalismo que los diagramas. Por otro lado, en [14] se ocupan de abordar la interoperabilidad entre los diagramas de secuencia y las RdP, a través de un enfoque basado en modelos. Esto se logra a través de un marco de transformación de modelo, que admite una transición entre estos modelos heterogéneos y permite la integración de diferentes conjuntos de herramientas. Esta combinación de interoperabilidad de modelos e integración de herramientas resulta en una reducción significativa de la complejidad en el desarrollo de software en general y en el análisis de modelos en particular. El modelo propuesto, se utiliza para modelar y analizar el comportamiento de una red de trabajo personal.

Siguiendo esta misma línea de correspondencia en [15-16], los autores proponen una línea similar a la propuesta anteriormente, con la salvedad de que, para determinar las transiciones entre los lugares, establecen un mayor de grado de formalismo matemático, ya que consideran la probabilidad de ocurrencia de cada una de las ramas o secuencias de los respectivos diagramas.

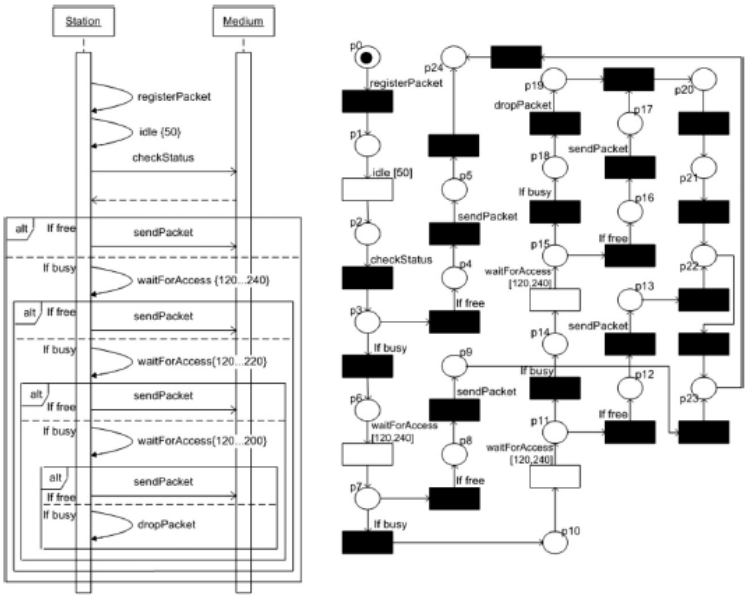


Figura 9 Comparación entre los diagramas de secuencia y las RdP [14]

Por otro lado, y siguiendo la línea de mapeo entre los diagramas de UML y las RdP, en [17], se establece la correlación entre los diagramas de Casos de uso y las RdP coloreadas, con el fin de formalizar los requerimientos y facilitar la validación y verificación de los mismos. Mientras que en [18], proponen un enfoque de trabajo para realizar una conversión de todos los diagramas de UML a RdP.

Un aspecto importante a la hora de desarrollar y poner en ejecución un software es la perfomance, cuanto mejor sea esta característica mejor uso tendrá por parte de los usuarios; y es por esto que, en [19] los autores desarrollan una metodología de trabajo que parte de los diagramas de UML, obteniendo un modelo intermedio denominado modelo de contexto de perfomance (PCM), para posteriormente obtener las RdP que permiten simular y estudiar la perfomance de los sitemas. El PCM permite realizar una descomposición modular del sistema de tal manera de facilitar su estudio, teniendo en cuenta las características relacionadas con la perfomance del mismo. Al poner a prueba este enfoque con una aplicación web de streaming de música, los autores concluyen que el mismo es eficiente para evaluar y predecir la perfomance de un sistema. Además, los autores determinar como punto de partida para futuros trabajos, el estudio de otras características de los sistemas software como ser confiabilidad y disponibilidad.

Por otro lado, en [20], los autores persiguen el objetivo de evaluar los requerimientos no funcionales de un sistema software. Para ello, consideran como punto de partida los diagramas de secuencia y los casos de usos, provisto por UML, para derivar los modelos necesarios para evaluar estos requerimientos. Estos modelos, se basan en la teoría de las RdP y mediante su simulación permiten evaluar que tanto satisface el diseño elaborado de un software, los aspectos no funcionales del mismo. Para ello, mediante fórmulas matemáticas derivan los modelos necesarios. El enfoque de trabajo propuesto, fue aplicado a

un sistema de red de cajeros de Irán, donde el mismo demostró su eficiencia para la resolución y evaluación de los requerimientos no funcionales del sistema bajo estudio. Si bien, el sistema estudiado es simple desde el punto de vista arquitectónico y de desarrollo, se destaca el hecho de poder evaluar la concurrencia y sincronismo de las tareas, lo que no descarta evaluar y estudiar sistemas de mayor envergadura, que es un punto de partida para investigación futuras de los autores.

En [21], los autores tienen como objetivo el estudio de técnicas que faciliten y mejores en el proceso de elicitación de requerimientos de los modelos teórico sistémico de análisis de accidentes. Para ello, en el trabajo se propone el uso de las RdP para analizar la interacción de los componentes de los sistemas mencionados y facilitar el relevamiento de los requerimientos. La aplicación del enfoque a un sistema de automóvil demuestra que funciona bien en componentes: Análisis de interacción y elicitación de requisitos de seguridad relacionados. Además, los autores proponen como objetivo futuro la integración del análisis de interacción de componentes con el análisis de seguridad del sistema normal con el fin de obtener requisitos de seguridad más completos, ya que el modelo de la RdP se puede traducir a un modelo matemático para los cálculos de probabilidad, se puede estudiar un análisis cuantitativo adicional de la seguridad del sistema con el nuevo modelo de causalidad.

En [22], los autores presentan una metodología de trabajo para obtener los correspondientes diagramas de UML a partir de los modelos expresados como RdP coloreadas. Para esto, los autores proponen la conversión a cada uno de los diferentes diagramas:

- **Casos de usos:** para la conversión a estos modelos consideran las transiciones y los lugares de las redes y realizan una conversión directa a caso de uso y actores, respectivamente.

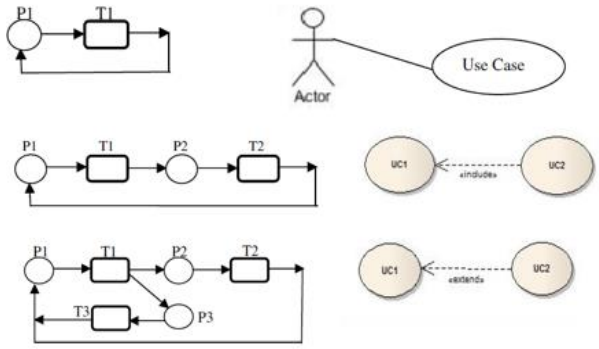


Figura 10 Conversión de RdP a CU [22]

- **Diagramas de Actividad:** para este caso los autores consideran los lugares definidos en las RdP para definir las acciones en el diagrama de actividades, posteriormente analizando el flujo de la RdP se determinan la conversión para las estructuras condicionales, cíclicas y las uniones de las actividades.

Los autores proponen la aplicación de este enfoque a un sistema de cajeros automáticos. Con esta aplicación, se facilita el proceso de validación de los requerimientos a partir de herramientas formales, que ayudan al proceso de desarrollo.

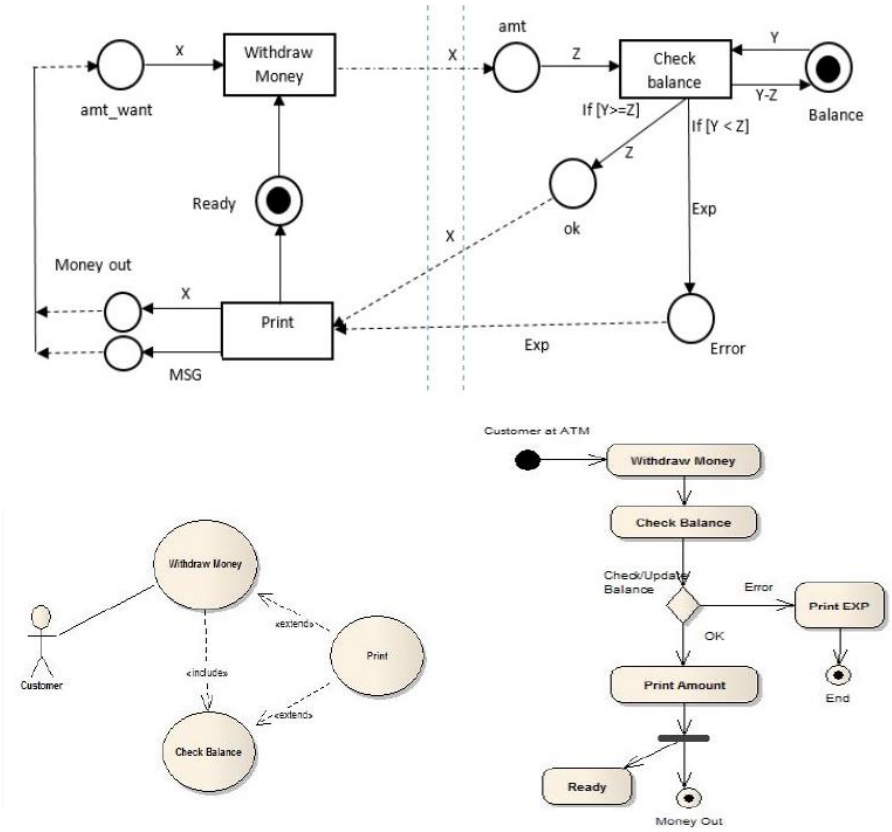


Figura 11 Conversión para el sistema de cajeros automáticos [22]

En [23], los autores diseñan un enfoque de trabajo que provee el mapeo entre los diagramas de análisis y diseño provistos por UML y los resultados provistos por el lenguaje ontología Web (OWL), para este mapeo los autores utilizan RdP coloreada. Este trabajo parte del hecho que los diagramas provistos por UML tienen un gran uso actualmente en el diseño de sistemas, son de visualización fácil, pero carecen de semántica del proceso de diseño de software, por otro lado, los lenguajes de ontología web explotan la semántica, pero no proveen una interfaz gráfica amigable. El enfoque propuesto y las herramientas desarrolladas son interesantes, al proyecto le está faltando herramientas que validen los resultados obtenidos, que es un enfoque que los autores están desarrollando como trabajo futuro a partir de la incorporación de dichas herramientas, a partir de la generación de archivos XSLT.

En los trabajos expuestos, se puede observar que el estudio del tópico de UML en relación con las RdP es un tema en auge y que presenta diversas facetas para evaluar el comportamiento de los sistemas y otorgarles mayor grado de formalismo al análisis y diseño obtenido por medio de los diagramas de UML.

Sin embargo, de este análisis del estado del arte, se pueden determinar otros aspectos que vinculan estas dos herramientas y que abarcan diversos aspectos del análisis, modelado y diseño de sistemas, buscando siempre obtener las mejores técnicas que ayuden a obtener un sistema más eficiente para el usuario final. Como se desprende, las RdP tienen un amplio abanico de aplicación, a continuación, se muestran diversos trabajos donde se aplican las RdP, que no necesariamente están vinculados con el desarrollo de softwares, sino que están relacionados con el modelado de sistemas discretos y estocásticos y su posterior simulación.

En [24], los autores proponen el desarrollo el uso de las propiedades y características de los sistemas de eventos discretos para modelar y analizar sistemas concurrentes, en especial evitar los deadlocks en los programas de memoria compartida, multihilo. Mediante el uso de una nueva clase de RdP, las redes Gadara; proponen su uso ya que permiten modelar explícitamente, programas multihilos con operaciones de bloqueo y adquisición de recursos. Además, los autores proponen algoritmos para medir, evaluar los resultados, comparándolos con las instancias de la literatura para OpenLDPA, que es una implementación open source para el protocolo de acceso de directorio ligero. Del análisis de los resultados los autores concluyen que los modelos desarrollados para evaluar los interbloqueos, resultaron exitosos y proponen continuar en el desarrollo de este tipo de RdP.

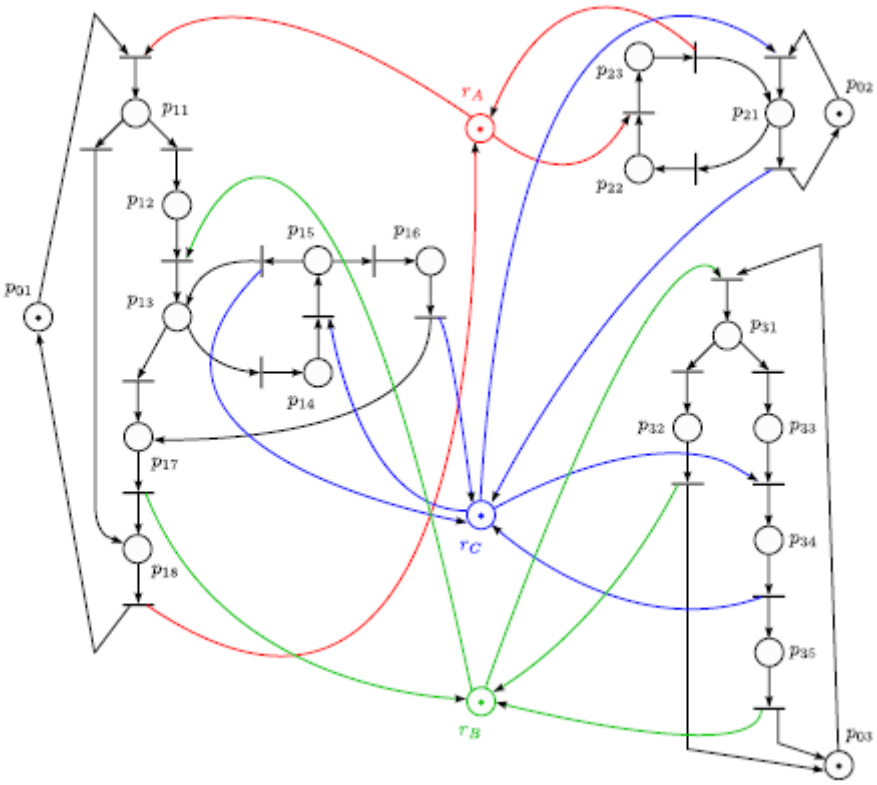


Figura 12 Modelo propuesto por los autores [24]

Siguiendo con la implementación de las RdP tipo Gadara, los autores en [25] buscan determinar el modelo óptimo que permita reducir el número de abrazos mortales en sistemas concurrentes y de esta manera lograr que el programa desarrollado trabaje de manera eficiente. Para ello, lo que proponen es un tipo de RdP denominado Gadara, su uso se basa en que Gadara se apoya en una base teórica rigurosa, que puede descomponer el objetivo práctico de eliminar los deadlocks en problemas formales bien estudiados, aprovechar un gran número de métodos probados y ofrecer garantías de seguridad/corrección y rendimiento. Los autores aplicaron el modelo desarrollado a diferentes ambientes multihilos, incluyendo un servidor apache, una aplicación cliente-servidor, entre otros, llegando a la conclusión de que la red planteada detecta correctamente los deadlocks, concluyendo que el modelo es un gran aporte a la literatura para resolver este tipo de problemas concurrentes, sin embargo, estaría interesante aplicar el enfoque similar para resolver otros problemas de concurrencia.

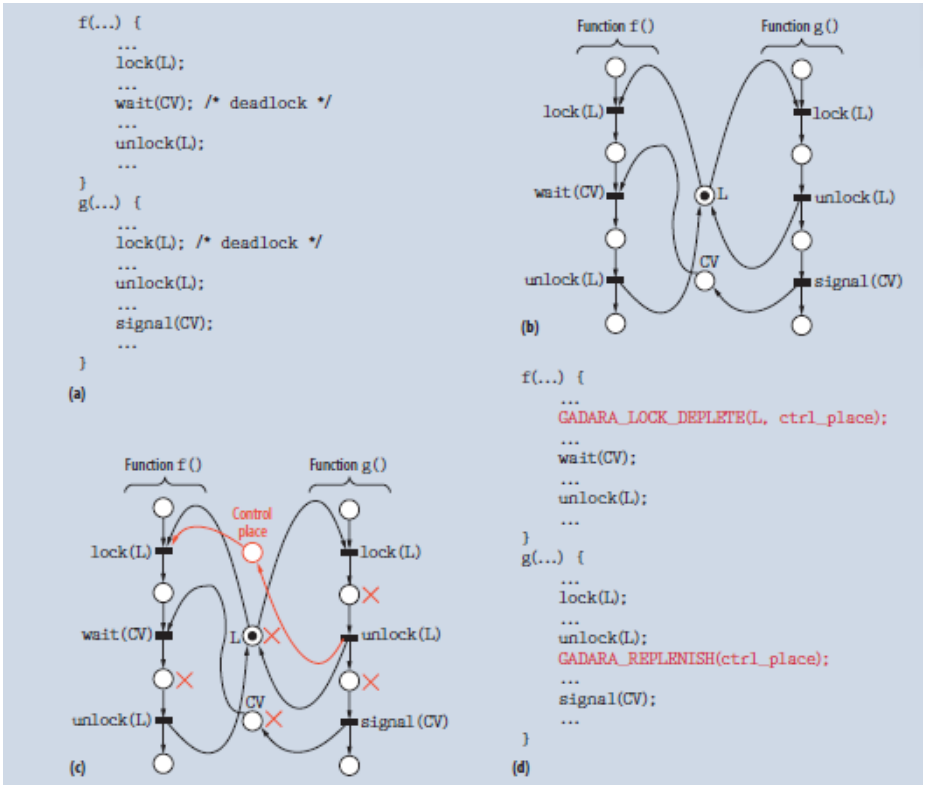


Figura 13 Modelo propuestos en [25]

Por otro lado, en [26] los autores proponen el uso de las RdP como medio para realizar el modelado de un controlador lógico. Para posteriormente, validar y verificar los requerimientos utilizando los diagramas de actividad propuestos por UML. Para ello, definen una serie de reglas a tener en cuenta en la conversión de RdP a los diagramas mencionados. Los autores concluyen que el modelado utilizando estas herramientas permite no solo determinar los requerimientos y validarlo más fácilmente, sino que ayuda a simular el controlador antes de la implementación.

Los autores, en [27], desarrollan un modelo matemático basado en RdP para poder simular y pronosticar la producción de crudo de petróleo en el corto plazo. Para ello, las condiciones de programación son derivadas de la capacidad del proceso, teniendo en cuenta la teoría de procesos. A partir de esto, los autores descomponen en subproblemas manejables y que son resueltos de manera jerárquica. Con esto, los autores desacoplan la interacción de eventos discretos y variables continuas. Estos subproblemas lo caracterizan en 3 aspectos fundamentales:

- **Modelado de los tanques:** considerando las restricciones de superposición de carga de los tanques.

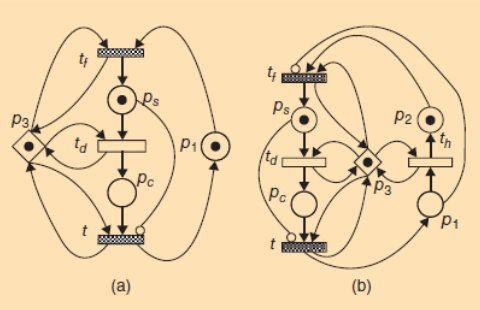


Figura 14 Modelo de carga de tanques [27]

- **Modelado de Tuberías**

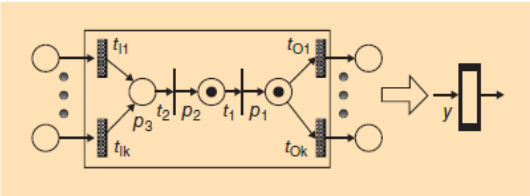


Figura 15 Modelado de tuberías [27]

- **Condiciones de programación de la producción:** que involucra las restricciones necesarias para la ejecución de la destilería para cada uno de los tipos de combustibles.

El modelo propuesto es aplicado a un caso de la industria petrolera de China. Los autores concluyen que los algoritmos desarrollados resultaron exitosos para las pruebas ejecutadas y además proponen las bases para futuras investigaciones en el marco de la resolución de problemas de programación de la industria de procesos, para desarrollar una herramienta de planificación industrial. Como así también, proponen la integración del modelo propuesto con las técnicas resientes y auge de Big Data.

En [28], se presenta una nueva metodología conocida como RdP orientada a la actividad (AOPN, por sus siglas en inglés). Esta, está diseñada para el modelando de sistemas discretos donde muchas actividades compiten por recursos comunes del sistema. AOPN deja el problema de gestión de recursos para ser manejado

por el sistema subyacente para que el foco este puesto en el modelado de las actividades. Además, muestran la utilización de dicha metodología al desarrollo de un sistema genérico. Los autores concluyen que la metodología presentada es una herramienta novedosa para el modelado de los problemas de gestión de recursos, cuando el mismo involucra actividades que demandan múltiple y genéricos recursos.

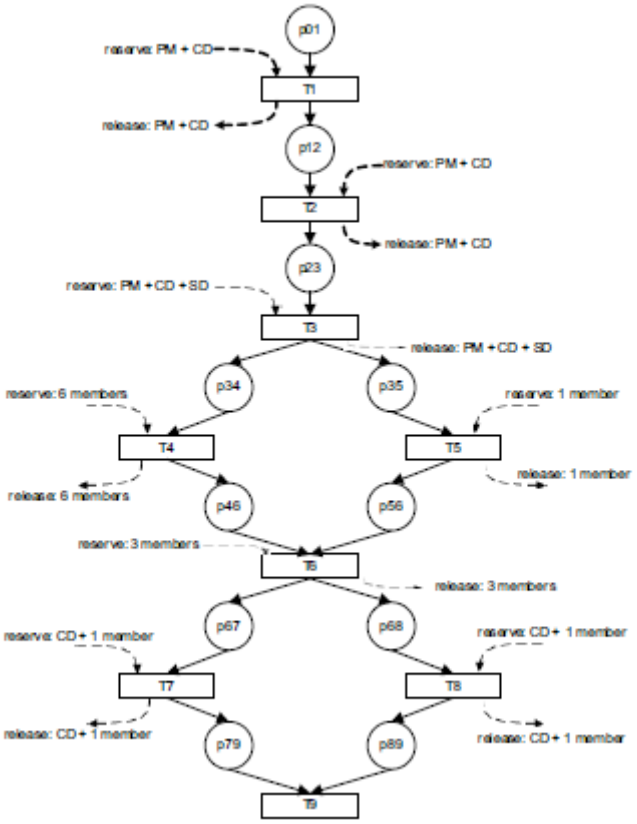


Figura 16 Modelo propuesto por los autores [28]

En [29], los autores proponen el desarrollo un enfoque que combina RdP con lógica difusa, con el fin de proporcionar herramientas para la investigación de procesos y sistemas biológicos. Esta herramienta, se formulan las hipótesis relacionadas con el tópico, permitiendo su posterior simulación y análisis. Los autores aplican el entorno a diversos problemas del mundo biológico, concluyendo que el enfoque desarrollado facilita el trabajo de las hipótesis y su simulación.

En [30], los autores desarrollan un modelo formal para la rutina de salida de una aeronave utilizando RdP como una herramienta formal de modelado y análisis para asegurar la optimalidad. El objetivo de este modelo es asegurar el correcto funcionamiento del sistema mediante la verificación de su comportamiento funcional, y que además, el modelo y el análisis propuestos aseguren la confiabilidad y seguridad del sistema al verificar que se cumplen con los requisitos de seguridad, como la separación entre aeronaves y la resistencia del proceso frente a condiciones desfavorables y riesgos potenciales como rutas de vuelo óptimas, planificación óptima de la secuencia, detección de conflictos,

planificación de salidas de aeronaves. Esto se realiza mediante el envío de señales de acuse de recibo entre los cuerpos de control y actuación (aeronave, controlador de rampa y controlador de puerta) como resultado de una comunicación exitosa. Los autores concluyen que el enfoque propuesto asegura la fiabilidad y seguridad del sistema bajo estudio. Dicha confiabilidad se ha logrado al verificar que el sistema cumple con los requisitos de seguridad (separación entre aeronaves y solidez del proceso frente a condiciones desfavorables y riesgos potenciales como rutas de vuelo óptimas, planificación de secuencia óptima, detección de conflictos, planificación de salida de aeronaves).

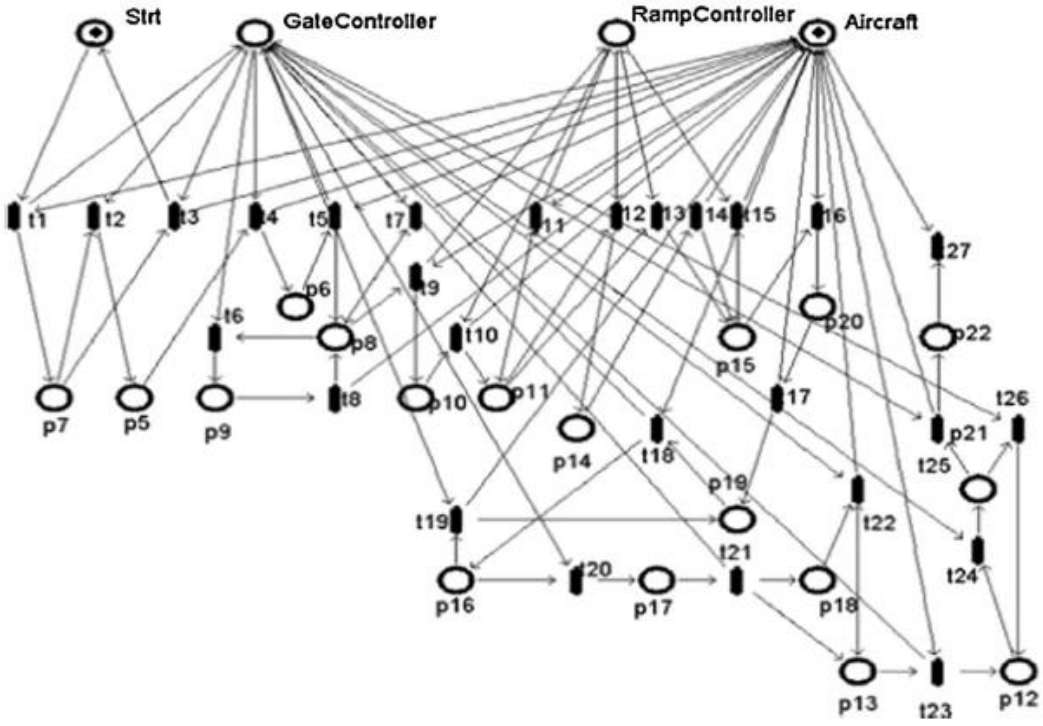


Figura 17 Modelo desarrollado por los autores en [30]

Los autores, en [31], realizan un estudio de la fabricación de los procesadores multi-core, poniendo énfasis en que la tolerancia a fallas de los mismos se convierte en un desafío, debido a que, en la fabricación de los procesadores, se utilizan tecnologías poco confiables. Es por esto que en el trabajo los autores utilizan como medio de prueba para la detección de los núcleos defectuosos, la técnica de prueba mutua, basada en pruebas aleatorias. El modelado y la ejecución de estas pruebas las llevan a cabo mediante el uso de las RdP estocásticas; y de los resultados los autores concluyen que la RdP arroja resultados dentro de los límites tolerables, en comparación a instancias de la literatura, y que la información suministrada por el modelo es de suma importancia para los ingenieros de procesos y optimizar y mejorar el proceso de manufacturación de los procesadores multi-core.

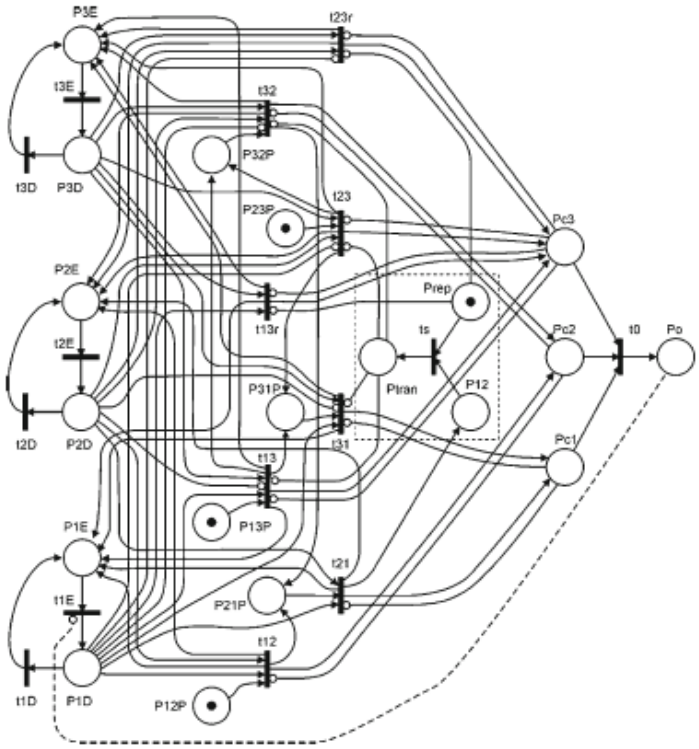


Figura 18 Modelado de una prueba mutua, para 3 procesadores [31]

En [32], se presenta un método sistemático para modelar e implementar el control en tiempo real para sistemas robóticos de eventos discretos utilizando RdP. En este trabajo, la especificación de la tarea de los procesos robóticos se representa como una red de nivel de control del sistema. Luego, según el enfoque jerárquico, se transforma en subredes detalladas, que se descomponen y distribuyen en los controladores correspondientes. Además, plantean la implementación del control distribuido en tiempo real a través de la comunicación entre el controlador del sistema y los controladores de la máquina en una red de microcomputadores se describe para un sistema robótico de muestra. A partir de la aplicación del método propuesto, los autores concluyen que es lo suficientemente general y se puede utilizar como una herramienta de creación de prototipos efectiva para un modelado y la simulación consistente, para los sistemas robóticos de tiempo real, más complejos.

El trabajo [33] se centra en el modelo completo de la RdP de color de un sistema de señalización ferroviaria típico de Tailandia: una estación de doble línea con un circuito de paso. El modelo incluye movimientos de trenes que se pueden simular y visualizar gráficamente. De acuerdo con los principios de señalización de los ferrocarriles estatales de Tailandia, en el trabajo se identificaron nueve propiedades fundamentales para el modelado y simulación de este sistema: enclavamiento de ruta, superposición alternativa, protección de flanco, secuencia de aspecto, liberación de señal de aproximación, bloqueo de aproximación, bloqueo de retroceso, liberación de ruta seccional y liberación de ruta rápida. A partir de los resultados, el autor concluye que el modelo propuesto arroja

resultados estadísticamente confiables, en comparación a un trabajo desarrollado anteriormente. Además, propone como trabajo futuro, analizar la escalabilidad y aplicabilidad de la técnica de barrido, para incorporar como mejora al modelo y obtener mejores resultados.

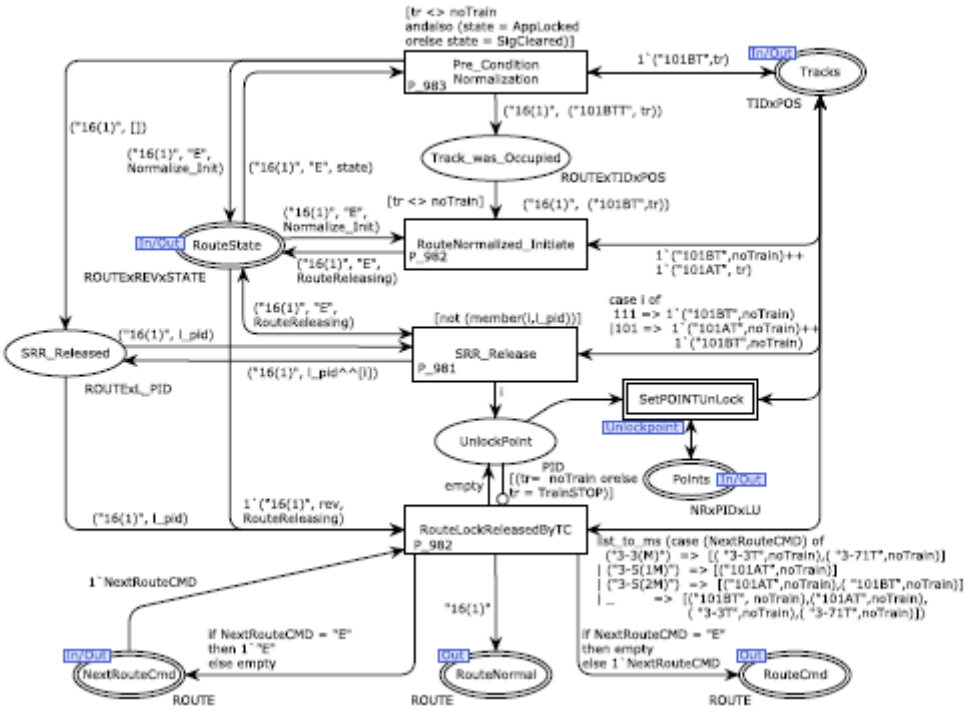


Figura 19 Modelo propuesto por los autores [33]

El artículo [34] se centra en analizar y estudiar modelos que permitan mejorar los procesos productivos de tal manera que la industria colombiana logre alcanzar metas de crecimiento significativas. Para ello, como un primer paso, consideran necesario concebir un modelo de agente autónomo que permita servir de patrón o módulo básico de tal forma que, por incrementos crecientes de complejidad, al estilo fractal, se pueda ir construyendo por niveles sucesivos de control el andamiaje total de una empresa, cualesquiera sean estructuras, procesos y productos. En base a estos supuestos, los autores justifican el uso de las RdP. A partir de los resultados obtenidos, los autores concluyen que se obtuvo una herramienta útil para visualizar y gestionar las etapas de proceso en forma sistemática a la hora de realizar integración la una con la otra. El Actor de Empresa es una herramienta eficaz, que sirve de una manera concisa a la hora de automatizar e integrar empresas a pequeña escala. La metodología aplicada en el artículo mostro excelentes resultados en el caso de estudio, pues proporciono de uno forma clara como se tenía que ir estructurando la realidad con la teoría.

El objetivo del trabajo [35] es presentar un método de asignación de abogados defensores a diversas causas, teniendo en cuenta la distribución equitativa de las mismas entre los abogados defensores de la administración pública y los abogados defensores externos; además de características propias de la causa, como ser tipo de crimen, distribución de la corte, características geográficas de la

región, entre otras similares. Para poder cumplir con este objetivo, los autores proponen un modelo dinámico, discreto y estocástico que combina las RdP, árboles de decisión y el método Monte Carlo. Como primer paso, los autores, simulan la cantidad de crímenes semanales, utilizando el método de Monte Carlo, para posteriormente, mediante el uso de las RdP determinan a que juzgado corresponde y que abogado defensor toma cada causa. A partir de los resultados obtenidos los autores concluyen que, aunque la cantidad de horas trabajadas diariamente por ambos tipos de abogados defensores es similar, la carga de trabajo anual es notablemente mayor para el personal de los abogados defensores, lo cual afecta directamente la calidad del trabajo. Sin embargo, modificando las condiciones de trabajo en el sentido de eliminar el límite de casos anuales en la licitación, aumentando el número de abogados defensores y la carga de trabajo, todos los abogados estarán bajo las mismas condiciones de trabajo.

En [36], los autores proponen la combinación de marcos y enfoque de trabajos para modelar sistemas distribuidos y concurrentes. Este enfoque aplica los conceptos, tecnologías y técnicas de las RdP y la teoría de agentes; a partir de la aplicación de un software orientado a agentes basados de las RdP, que permite el paradigma multiagente y aplica como lenguaje de implementación, las redes mencionadas. Los autores, en [37], plantean los inconvenientes del control bidimensional del tren a lo largo del interruptor y el paso a nivel. Para poder suplir estos inconvenientes, se integran los conceptos de agente móvil con los de RdP y desarrollan un enfoque de RdP móvil (MPN) que aborda tanto la movilidad como la concurrencia. Al aplicar los conceptos de MPN al bloque móvil que se entrelaza a lo largo del interruptor y el paso a nivel para llevarlo a un control dimensional. A partir de la ejecución del modelo, los autores obtienen como conclusión que el modelo de MPN resulta exitoso, ya que permite capturar la concurrencia, la secuencialidad y la movilidad del sistema de bloqueo de trenes a lo largo del interruptor y el paso a nivel. También concluyen que, se reduce la complejidad del sistema y se facilita la visualización de las propiedades importantes necesarias para la fase de implementación.

En [38], los autores toman los conceptos de las RdP coloreadas (CPN) y extienden los conceptos de las RdP al modelado de interacción web. Posteriormente, los autores proponen la aplicación de estos conceptos desarrollados a la interfaz web de *Classroom Experiencie*, una plataforma educativa. Además, agregan las capacidades de temporización con el fin de evaluar el tiempo promedio que los usuarios tienden a pasar durante las interacciones. De los resultados obtenidos, los autores concluyen que el enfoque propuesto representa fácilmente los diferentes niveles de acceso entre los usuarios. También, compararon los resultados del enfoque desarrollado con los diagramas de actividad y casos de uso provistos por UML, observando que reflejan la misma información utilizando menos recursos y modelos; lo que permite concluir que el trabajo arroja resultados prometedores.

El objetivo del trabajo propuesto en [39], consiste en determinar una forma de desarrollo web eficiente a partir de la ejecución de pruebas exitosas. Para ello, consideran el modelo de la aplicación web a través de una RdP, que permite evaluar el comportamiento del sistema, simularlo y analizar los resultados. Esto lo logran a partir de realizar un análisis de cobertura del código con el fin de determinar el modelo óptimo de aplicaciones web que involucran sesiones basadas en el navegador HTTP. Este modelo fue evaluado en 18 aplicaciones web, generando 857 pruebas las cuales detectaron 57 bugs, que comparando estos resultados con la ejecución de pruebas realizadas por testers, que, si bien detectaron más casos de prueba, solo pudieron encontrar un tercio de los bugs detectados por las pruebas generadas por el modelo; esto le permite concluir a los autores que su enfoque de trabajo produce resultados eficientes.

En [40], los autores abarcan el estudio de uno de los aspectos importantes en el desarrollo de cualquier sistema, el estudio de los riesgos asociados a las restricciones de tiempo. Para ello consideran el uso de RdP temporalizadas para evaluar un sistema de tiempo real y determinar el comportamiento y obtener conclusiones. El modelo desarrollado, se basa en determinar la probabilidad de ocurrencia de los riesgos y las consecuencias que puede llegar a producir en el sistema, a partir de esto, y considerando los modelos de diseño provistos por UML, los autores obtienen las fórmulas matemáticas que le permiten deducir las RdP. A partir de la ejecución de estas redes en un sistema real, concluyen que, si bien las redes son lentas en el procesamiento, los resultados obtenidos son satisfactorios.

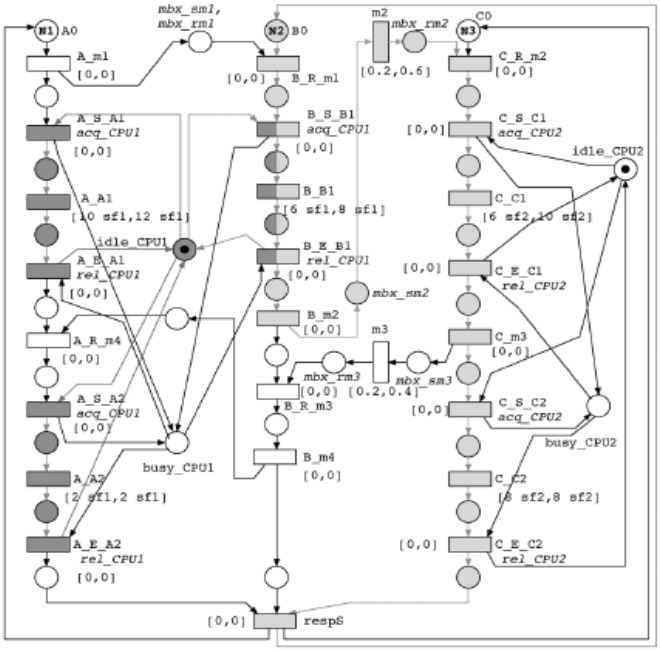


Figura 20 Modelo propuesto por los autores [40]

Un aspecto fundamental en el desarrollo de cualquier sistema software, son la ejecución de las pruebas para validar y verificar los resultados arrojados por el programa que se está desarrollando. Por eso, en el trabajo propuesto en [41], los autores proponen el desarrollo una herramienta para generar test automáticos, a partir de RdP. En este trabajo, a partir de las especificaciones brindadas por el usuario, convierte las mismas en modelos de RdP que permiten ejecutar las pruebas para detectar posibles bugs que puedan existir en el sistema. Los autores ejecutaron estos modelos con diferentes programas y la herramienta demostró su eficiencia en el desarrollo y ejecución de los casos de pruebas, en los procesos de desarrollo incremental. Sin embargo, esta herramienta no prevé pruebas para entornos concurrentes, lo que está marcado como una línea de trabajo futura por los autores.

En [42], los autores proponen una aplicación de las RdP para poder simular un sistema de eventos discretos. En este caso, los autores desarrollan un enfoque de trabajo que lo denominan GPenSim que se basa en una descomposición modular del desarrollo de modelos basados en RdP; ya que consideran que toda la información de la o las redes se encuentra almacenadas en archivos, distinguiendo los aspectos dinámicos y estáticos de las redes. Para posteriormente, introducirlos en una capa que se encargue de procesarlas y simularlas para posteriormente, mediante la conexión con una capa externar mostrar los resultados. Este modelo es usado para simular un modelo que evalúa el tráfico aéreo en el aeropuerto de Evenes, en Noruega. De la ejecución del mismo, los autores concluyen que el enfoque desarrollado es robusto y funciona para el modelado de diversos sistemas de eventos discretos, sin embargo, no consideran los pesos de los arcos como variables durante la ejecución de la simulación y podrían considerar en vez de un reloj global, considerar el uso de los timers provistos por las computadoras aportando más realismo a las simulaciones, sobre todo en sistemas de este estilo.

El artículo [43], propone una técnica que emplea las herramientas de la RdP para modelar, simular, analizar y controlar una casa inteligente. Para ello, dividieron la casa en módulos y para cada uno de ellos, desarrollan el modelo correspondiente, estos módulos fueron: Sistemas de incendio, Control de la temperatura y Acceso remoto. De la simulación del modelo, los autores concluyen que el mismo es efectivo para el modelado de los hogares inteligentes, ya que incluye escalabilidad, extensibilidad y manejo de la concurrencia. En [44], los autores consideran que los mecanismos para la provisión de datos en los Data Warehouse presentan dificultades e introducen errores, que ocasionan que los costos de corrección sean elevados, entonces proponen un RdP coloreada con el fin de modelar el aprovisionamiento de datos de tal manera de reducir errores.

Por otro lado, en [45] desarrollan un modelo de RdP lógica para poder modelar y simular los sistemas relacionados con el comercio electrónico. El modelo desarrollador por los autores se presenta en la figura 20.

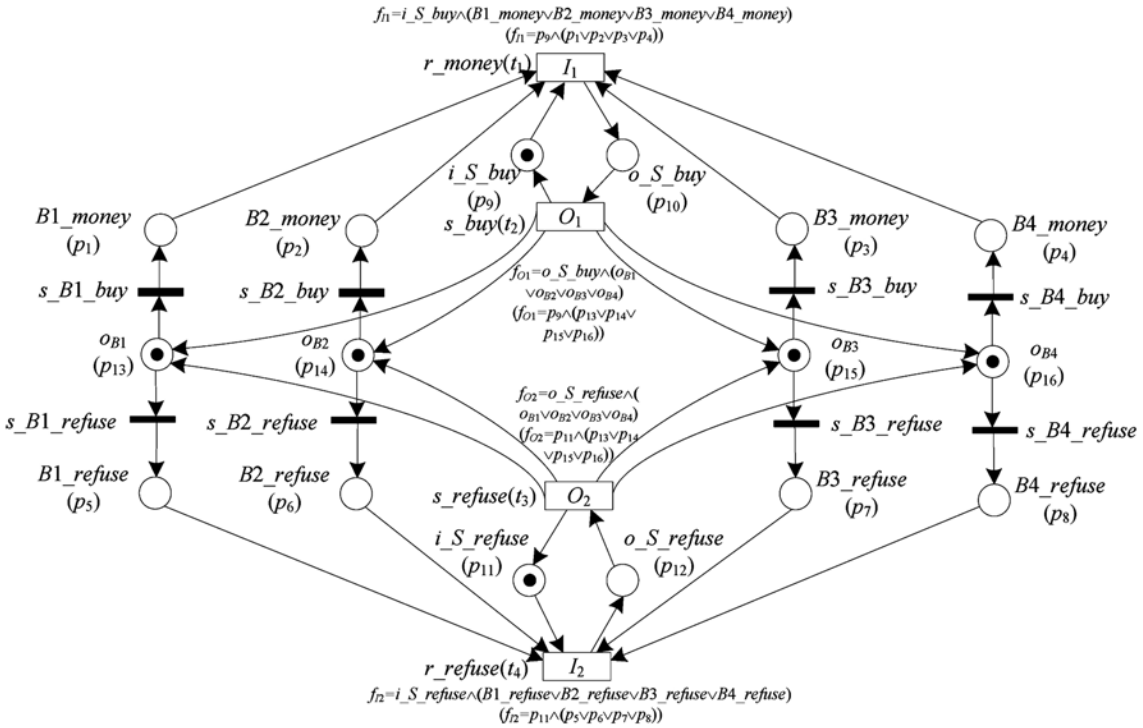


Figura 21 Modelo presentado por los autores [45]

Además, en [46] presentan una aplicación de las RdP para optimizar los sistemas relacionados con el control en los procesos de manufacturación. Por otro lado, en [47], los autores desarrollan un modelo basado en RdP para los sistemas que utilizan un protocolo de compartir tareas en sistemas multiagentes, que se encuentran ubicados a lo largo de una red. Diseñan una RdP coloreada con el fin de determinar a tiempo las muertes de los agentes contratistas de la red y lograr reducir el número de fallas del sistema total. También, en [48], se presenta un modelo basado en RdP para modelar el sistema de control de trenes en China y Alemania. Por último, en [49], los autores desarrollan un modelo para estudiar los sistemas de reacción – difusión, utilizando RdP coloreadas.

A partir del estudio bibliográfico, se puede concluir que las RdP poseen un amplio espectro de aplicación en cualquier ámbito de la vida cotidiana, produciendo resultados satisfactorios que logran mejorar los procesos y las líneas de investigación con el objetivo de lograr mejores resultados a futuro. En cuanto a la ingeniería de software, las RdP se encuentran fuertemente vinculadas con el proceso, ya que ayudan a formalizar los procesos de validación y verificación de requerimientos como así también todas las etapas del proceso de desarrollo de sistemas y lograr así resultados más eficientes.

Capítulo 3: Enfoque propuesto

En este capítulo se presentará el enfoque propuesto, conjuntamente con los fundamentos teóricos que subyacentes a los desarrollos propuestos. Este capítulo se divide en dos partes, en la primera se presentan las características de los sistemas discretos, dinámicos y estocásticos y su modelado.

El objetivo de esta tesis consiste en complementar el proceso de análisis y diseño de sistemas, para ello, en la segunda sección de este capítulo se propone la integración de las RdP etiquetadas con los diagramas que propone UML. Conjuntamente, se detallan los fundamentos teóricos que llevaron a determinar los pasos para convertir del diagrama de UML a las RdP.

3.1 Modelado de Sistemas Discretos, Dinámicos y Estocásticos

Las características usuales de estos tipos de sistemas, como ser concurrencia, toma de decisiones, sincronización y prioridades, fácilmente pueden ser modeladas por las RdP.

1. ***Ejecución Secuencial:*** Este tipo de características de los sistemas dinámicos, donde una acción se ejecuta después de la otra, puede ser modelada por una RdP estableciendo la condición de que la transición t_2 solo se dispara si la transición t_1 de disparó previamente. Un ejemplo de esto, se visualiza en la figura 22.

Figura 22 Ejecución Secuencial

2. ***Conflictos:*** Este tipo de modelado se observa en la figura 23, donde la transición t_2 y t_1 se encuentran en conflicto ya que si bien ambas transiciones se encuentra habilitadas el disparo de una de ellas, deshabilita la otra. Para poder resolver este conflicto se puede hacerlo de una manera no probabilística o bien asignando valores de probabilidad a cada una de las transiciones conflictivas.

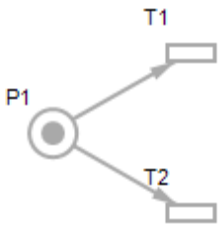


Figura 23 Conflicto

3. ***Concurrencia:*** Es un factor fundamental en los sistemas bajo estudio y el modelado de esto, se observa en la figura 24.

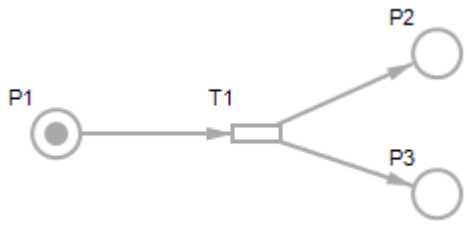


Figura 24 Concurrencia

4. ***Sincronización:*** Aspecto fundamental de los sistemas de interés de esta tesis. En este caso se considera una transición t_1 conectada a los lugares y esta transición solo se dispara cuando ambos lugares ha recibido *tokens*.

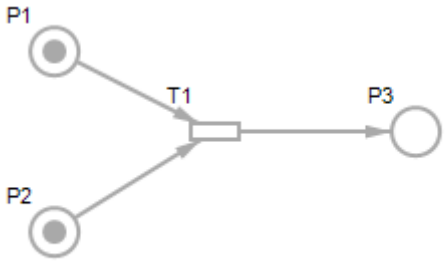


Figura 25 Sincronización

5. ***Exclusión Mutua:*** En la figura 26 se muestra el modelado, utilizando RdP, para esta propiedad. En este caso, dos procesos no pueden ejecutarse simultáneamente debido a las restricciones de uso de los recursos asociados.

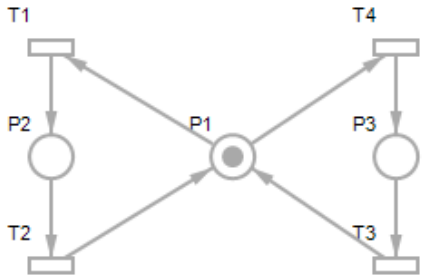


Figura 26 Exclusión Mutua

6. ***Prioridades:*** las definiciones clásicas de una RdP no contienen un mecanismo de representación de prioridades. Sin embargo, las mismas se pueden adaptar de tal manera de incluir estas situaciones, para ellos se utilizan los arcos inhibidores. Este conecta un lugar a una transición y se caracteriza por que en un extremo contiene un círculo. La presencia de este arco cambia las condiciones de habilitación de una transición, ya que para estos casos una transición se considera habilitada si cada lugar de entrada

conectada a la transición por un arco normal contiene al menos un número de *tokens* igual al peso del arco, sin que haya *tokens* presentes en cada lugar de entrada conectado a la transición por un arco inhibidor. La regla de disparo de la transición es la misma para los lugares normalmente conectados, con la salvedad que el disparo no cambia el marcado en los lugares conectados mediante un arco inhibidor.

En la figura 27, la transición t_1 está habilitada si el lugar p_1 contiene un *token*, mientras que t_2 está habilitada si p_2 contiene un *token* y p_1 no contiene ninguno. Esto da una prioridad mayor a t_1 sobre t_2.

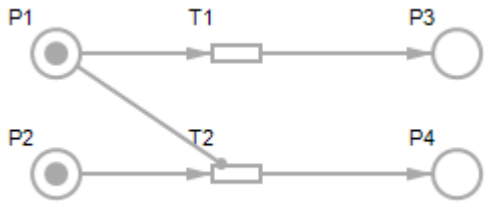


Figura 27 Prioridad

3.1.1 El Tiempo en las Redes de Petri

Uno de los aspectos críticos de los sistemas estudiados en este trabajo, es el paso del tiempo, necesario para la simulación de modelos dinámicos. Originalmente las RdP fueron propuestas como un modelo casual sin ninguna noción de tiempo o de probabilidad; sin embargo, para muchas aplicaciones el paso del tiempo es una necesidad. Para considerar incorporar el tiempo a las redes es necesario analizar los siguientes aspectos:

1. ***Localización de retrasos:*** existen diversas maneras de introducir el tiempo en los modelos de RdP. Como se dijo una RdP consiste en un conjunto de lugares y transiciones conectadas mediante arcos; por lo tanto, el tiempo puede ser asociado a cualquiera de estos elementos. En la mayoría de los modelos de RdP temporalizadas, las transiciones determinan retrasos de tiempo. Solo en algunos modelos, los retrasos de tiempo son determinados por los lugares y/o arcos; ya que parece más natural asociar el tiempo a las transiciones, pues antes todo, estas representan actividades y son estas últimas las que consumen tiempo. Sin embargo, existen autores que argumentan que es más conveniente asociar el tiempo a los lugares puesto que esto deja intacta a la regla de disparo original, es decir, que la habilitación y el disparo son instantáneos.

2. ***Tipos de retraso:*** independientemente donde coloquemos los retrasos, se pueden distinguir distintos tipos de los mismos, estos pueden ser determinísticos o estocásticos. Los primeros permiten métodos de análisis simples, pero aplicaciones limitadas. Cuando consideramos aplicaciones del mundo real, los retrasos corresponden a la duración de las actividades

que son típicamente variables. En la mayoría de los modelos de RdP temporalizadas, los retrasos se describen mediante distribuciones de probabilidad.

3. ***Semántica de preselección y Semántica de competencia:*** Al agregar el tiempo a las RdP, las reglas de habilitación y disparo necesitan ser modificadas para especificar cómo los conflictos son resueltos (la relación entre habilitación y disparo) y si el disparo es instantáneo o no. Claramente, estos dos aspectos están relacionados. Suponiendo que las transiciones determinan los retrasos, si el disparo es instantáneo (es decir, no toma ningún tiempo), entonces es necesario asociar el tiempo a la habilitación de una transición. Si el tiempo está asociada a la habilitación, no hay ninguna necesidad de definir explícitamente como son resueltos los conflictos. Es decir, las transiciones habilitadas compiten entre sí y la que es programada para disparase primero, lo hará. Esto es lo que se conoce como **semántica de competencia.**

   También es posible especificar explícitamente la manera en que los conflictos se resuelven, esto es lo que se denomina **semántica de preselección.** En esta última, no hay competencia alguna entre las transiciones habilitadas; en el momento en que las transiciones se vuelven habilitadas, una de ella es seleccionada. Habitualmente, la semántica de competencia se combina con el disparo instantáneo, por lo tanto, se utiliza el término de retraso de habilitación para referirse a esta semántica.

   Generalmente, la semántica de preselección, se combina con tiempo de retención, es decir, que los *tokens* residen por cierto tiempo dentro de un lugar o transición. Notar que, para la semántica de competencia, la resolución de conflictos y el retraso se maneja por el mismo mecanismo. Mientras que, para la semántica de preselección, el mecanismo para resolver conflictos es separado del retraso actual. Muchos modelos de RdP estocásticas usan la semántica de competencia. La misma permite una traducción más directa a cadenas de Markov y es más expresivo, mientras que la semántica de preselección es más intuitiva y más fácil de usar.

   Para la semántica de preselección, los retrasos (tiempos de retención) poder ser asociados al disparo de la transición, o al tiempo mínimo que un *token* pasa en un determinado lugar. Mientras que, para la semántica de competencia, los retrasos se asocian al tiempo de habilitación, puede ocurrir que una transición sea deshabilitada por otra en caso de conflicto. En tal caso, la primera transición pierde la competencia y no se disparará. Si la transición se vuelve a habilitar, una nueva competencia comienza.

4. ***Políticas de Memoria:*** Como se explicó anteriormente, puede ocurrir que una transición quede deshabilitada por el disparo de una transición con la

que queda en conflicto. Entonces para definir como queda el reloj de una transición cuando se vuelva a habilitar, se definieron dos mecanismos básicos:

- ***Continuar:*** la transición deshabilitada "para" la cuenta atrás del reloj y la continua la próxima vez que la transición es habilitada.
- ***Reiniciar:*** la transición reinicia su reloj (quizá con un nuevo valor) cada vez que la transición es habilitada.

Estos mecanismos definen las políticas de memorias de una transición:

- ***Memoria de Envejecimiento:*** el tiempo de habilitación restante se congela en el momento que la transición se vuelve inhabilitada y se reanuda en el momento que la transición se vuelve a habilitar.

- ***Memoria de Habilitación:*** un tiempo nuevo de habilitación se muestra cada vez que una transición se vuelve habilitada, es decir, que las transacciones antes interrumpidas tienen que empezar desde el principio.

- ***Memoria de Reajuste:*** un tiempo nuevo de habilitación muestra cada vez que una transición se dispara, es decir, también las transiciones no en conflicto con la transición en disparo son interrumpidas y tienen que relanzar un tiempo nuevo de habilitación.

5. ***Capacidad, prioridad y política de colas:*** para las RdP temporalizadas, la capacidad de los lugares y las transiciones es relevante. Los lugares pueden tener una capacidad limitada para restringir el número de *tokens* residiendo en un lugar al mismo momento. Las transiciones pueden tener capacidad para limitar el número de disparos concurrente de la misma transición. A partir de esto, se pueden identificar tres semánticas de capacidad:

   - ***Semántica de Servidor Simple:*** la capacidad de un lugar o una transición es uno.

   - ***Semántica de Servidor Múltiple:*** la capacidad de un lugar o una transición es un número entero positivo mayor que uno.

   - ***Semántica de Servidor Infinito:*** en este caso no existen restricciones de capacidad.

   Muchos modelos de RdP temporalizadas, permiten un mecanismo de prioridad, es decir, si múltiples transiciones compiten por el mismo *token*,

la transición con la prioridad más alta dispara; este mecanismo puede servir de ayuda para la semántica de preselección. En los modelos ampliamente utilizados de estos tipos de RdP, las transiciones inmediatas tienen prioridad sobre las transiciones temporalizadas.

Algunos modelos de RdP permiten a especificación de políticas de colas. Puesto que los *tokens* en el mismo lugar, y del mismo color, son indistinguibles, a menudo no tiene sentido elegir una disciplina de cola.

3.2 Redes de Petri y UML

Al momento de realizar el análisis y diseño siguiendo utilizando UML, se desarrollan diversos diagramas que permiten comprender el funcionamiento del sistema y de esta manear facilitar su codificación. Sin embargo, para los sistemas discretos, dinámicos y estocásticos con los diagramas que proporciona UML no es suficiente para obtener un entendimiento global del sistema; es por esto que, en esta sección se detallan las relaciones que existen entre los diversos diagramas de UML y las RdP con el fin de desarrollar las herramientas que permitan complementar ambas herramientas y de esta manera lograr una más robusta.

A partir de esto, el objetivo del trabajo se centra en poder traducir los principales modelos que propone UML en una RdP estocástica etiquetada, con el fin de poder visualizar claramente las distintas características de los sistemas estudiados en este trabajo de tesis, sobre todo las restricciones de tiempo asociadas a cada transición y/o estado.

3.2.1 Diagramas de Estados

Para realizar el estudio de la conversión de determinaron los siguientes componentes de los diagramas: estados (incluyendo el estado inicial y final del diagrama), transiciones y pseudo-estados de elección, como se observa en la figura 28, lo que permitió modelar y evaluar una amplia gama de problemas cuantitativos, incluido el estudio de caso presentado.

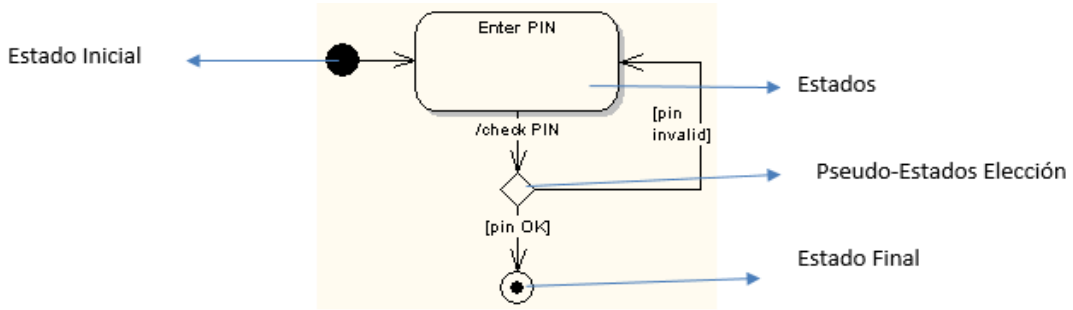


Figura 28 Modelo genérico de Diagrama de Estados

3.2.1.1 Estados

Cuando analizamos los estados de los diagramas podemos observar que existen siempre tres tareas asociadas a cada uno de ellos (ingresar al estado, hacer una determinada acción y salir del estado), además cada una de estas tareas puede o no tener un tiempo asociado, que determina el retraso de la misma. Todo esto nos da el puntapié inicial para poder comenzar con la transformación desde los diagramas antes mencionados a la RdP.

Independientemente de si hay actividades opcionales especificadas para un estado, siempre seguimos el orden lógico y temporal de las actividades opcionales. Se crea una transición inmediata en la traducción de la RdP para cada actividad. Por lo tanto, la traducción de un estado X siempre da como resultado un lugar que representa el inicio a la actividad, acompañado por la transición que simula el acceso al estado. Seguido de esta última se agrega nuevamente un par lugar, transición que se encargan de modelar la acción que se ejecuta dentro del estado, esta transición puede tener asignado un tiempo o no dependiendo si la misma lo necesitará. Por último, se añade un nuevo par que se encargan de representar la salida del estado que se está modelado, desde este último lugar, se añaden las transiciones apropiadas para cada posible estado posterior.

3.2.1.2 Transiciones

El otro aspecto importante a la hora de graficar una RdP son las transiciones de las mismas, una parte ya de discutió en la sección anterior. Para complementar la construcción de las transiciones, es necesario tener en cuenta que se puede consumir tiempo dentro de cada estado durante la ejecución de las actividades opcionales de entrada, ejecución y salida.

Es por esto que a cada transición se le asocia el tiempo necesario para realizar la misma, este tiempo viene asociado por una etiqueta que se le asocia a cada transición indicando el tiempo necesario para poder cumplir dicho movimiento y poder avanzar entre los diversos estados. Estos tiempos pueden ser valores constantes, como valores continuos, por ejemplo, valores que provienen de una determinada distribución, con los parámetros necesarios.

3.2.1.3 Bifurcaciones

Además de los estados, la transición y las anotaciones de tiempo, otro elemento importante de los sistemas de eventos discretos estocásticos son las elecciones no deterministas. Para esto, se define un pseudoestado que actúa como un selector que determina cual estado se debe ejecutar en función de que ocurra una determinada acción.

Como es imposible, sin especificar matemáticamente, cual estado debe ejecutarse, es necesario establecer valores de probabilidad para cada uno de los posibles caminos que pueden tomar desde un pseudoestado.

3.2.1.4 Estado Inicial y Final

El estado inicial de un diagrama de estado, es un lugar que contendrá con los marcadores iniciales de la red. Mientras que, para el estado final, se definirá un lugar.

3.2.1.5 Algoritmo de Conversión

A modo de resumen, se puede expresar en términos de pseudocódigo lo expuesto anteriormente, como sigue:

Datos de Entrada: *Diagrama de transición de estados (1)*
Datos de Salida: *RdP Estocástica Etiquetada (2)*

(* Conversión de los estados*)
Para cada Estado (s) de (1) hacer
    Crear los lugares que simulen el ingreso, la ejecución y la salida de la actividad
    Crear la transición que una cada uno de los lugares anteriores
    Establecer el tiempo de demora para cada una de las transiciones

(* Para cada uno de los pseduoestados de elección*)
Para cada uno de los estados de decisión (d) de (1) hacer
    Crear el estado *LugarEleccionD*
    Para cada una de las transiciones de salida de D hacer
        Crear el lugar de destino g
        Crear la transición del lugar g a s
        Establecer el peso de la transición

Crear un lugar con los tokens necesarios que represente el inicio de la red
Crear las transiciones hacia las actividades correspondientes
Crear un lugar que represente el estado final de la red
Crear las transiciones de las actividades hacia dicho lugar

Figura 29 Algoritmo de conversión de Diagramas de Estados a RdP

3.2.2 Diagramas de Actividad

Una Actividad es el modelo del proceso, representado en un Diagrama de Actividad por un cuadro grande que contiene nodos y bordes. Una actividad consta de varios nodos de acción (también llamados nodos ejecutables, más precisamente acciones opacas), denotados por bloques redondeados más pequeños; cada uno representa alguna acción que se realiza durante algún tiempo durante este proceso. Solo nos interesará el tipo más común de bordes de

actividad: los bordes de flujo de control conectan acciones para definir su secuencia de ejecución. Esta secuencia comienza en el nodo inicial y termina en el nodo final de flujo. Estas dos no son acciones reales, sino nodos de control especiales.

Los nodos de control adicionales incluyen el nodo de decisión, que representa una ramificación en dos o más rutas posibles de ejecución de acuerdo con algún criterio. Estas rutas alternativas se encuentran en los nodos de control denominados nodos de fusión, desde qué punto proceden con la misma secuencia. Un par similar de nodos de control es división y unión, que denotan respectivamente un punto donde la ejecución se divide en varios subprocesos concurrentes, y un punto donde estos subprocesos de eventos se unen entre sí para proceder como una única secuencia.

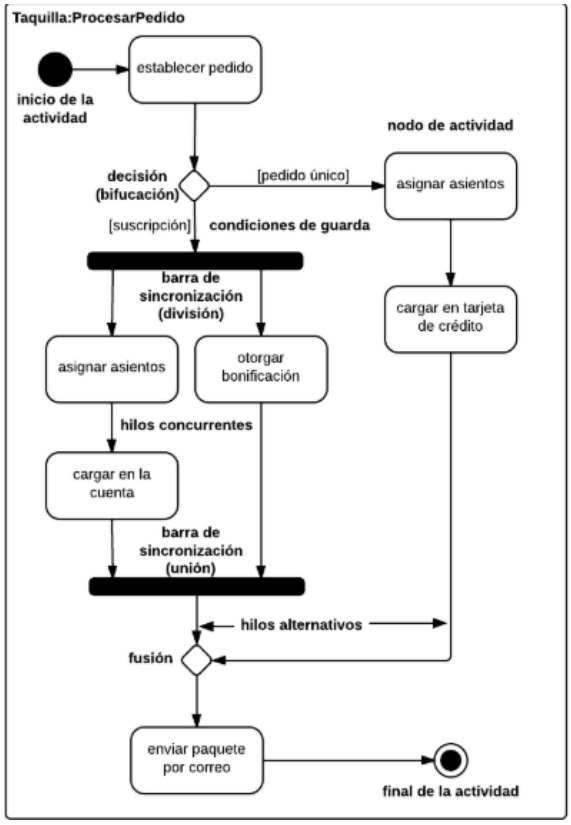


Figura 30 Elementos a tener en cuenta de un diagrama de actividad [51]

Entonces para poder obtener la RdP equivalente a un diagrama de actividades, debemos considerar las traducciones de los siguientes elementos.

3.2.2.1 Nodos de Acción y flujo de control

A partir de la interpretación de la semántica de la actividad UML de forma tal que la ejecución y la acción no son instantáneas, pero no pueden durar un tiempo muy prolongado. Además, no hay ninguna obligación de que una acción comience tan pronto como finalice la anterior, puede pasar algún tiempo entre ellas. Por lo tanto, para cada Acción, la RdP resultante debe tener uno o más

estados donde se está ejecutando la acción. Del mismo modo, para cada límite de flujo de control, debe haber estados de la RdP que indica que la ejecución de la acción ha terminado la primera acción, pero no ha comenzado la siguiente.

Este objetivo se puede lograr creando un lugar para cada acción y cada flujo de control del modelo de Actividad. En el marcado asociado con un estado de RdP, un token en el lugar correspondiente a una acción significa que una instancia del proceso está ejecutando la acción actualmente. De manera similar, un token en el lugar corresponde a un flujo de control significa que el límite de actividad está actualmente activo.

Obviamente, también necesitamos transiciones, de lo contrario, la RdP sería estática. Para cada flujo de control entrante en una acción, se debe crear una transición para representar el inicio de la ejecución de la acción. Esta transición tiene un arco entrante desde el lugar que representa el flujo de control y un arco saliente hacia el lugar que representa la acción. Por lo tanto, la transición se habilita si el límite de la actividad está activo, y al activarlo iniciará la ejecución de la acción y cesará la activación del límite. De manera similar, una transición debe representar la terminación de la acción, con un arco entrante desde el lugar de la acción y un arco saliente hacia el lugar del borde.

Este argumento sugiere el siguiente principio para la transformación. Cada límite de flujo de control se asignará a un lugar que representa cuando el límite está activo. Cada acción debe adaptarse en dos transiciones y un lugar intermedio, con dos arcos que unen los tres nodos. Las primeras representaciones de transición que entran en la acción, el lugar representa que el estado se encuentra ejecutando la acción, y la segunda transición representa salir de la acción. Luego, la introducción de la acción se debe vincular mediante un arco entrante al lugar que representa el borde de actividad entrante, y la transición de salida debe vincularse mediante un arco saliente al lugar que representa el borde de actividad saliente.

3.2.2.2 Nodo Inicial y Final

Cuando la implementación alcanza un nodo final de flujo en la Actividad, la instancia del proceso finaliza. Más precisamente, si hay varios subprocesos de ejecución, debido a un nodo de bifurcación, el que alcanza el nodo final de flujo termina, mientras que otros no se ven afectados. En la terminología de la RdP, el nodo final del flujo debe consumir el token que representa la activación del límite de flujo de control entrante, sin colocar un token en ningún lugar que represente un borde o acción de actividad. Esto se logra mediante una transición que tiene un arco entrante desde el lugar correspondiente al límite del flujo de control que apunta al nodo final del flujo.

Hay al menos dos formas de interpretar un nodo inicial. Una posibilidad es que el nodo pueda activar el límite del flujo de control saliente exactamente una vez,

iniciando así el proceso; sin embargo, en algunos contextos, tiene más sentido abarcar que el nodo inicial puede activar el flujo de control un número arbitrario de veces, iniciando varias instancias de proceso que se ejecutan simultáneamente.

Ambas versiones pueden ser representadas por RdP, aunque en el segundo caso habría un problema semántico con nodos de unión, que está fuera del alcance de este tutorial. En ambos casos, el nodo inicial se asigna en una transición, con un arco saliente al lugar correspondiente al límite del flujo de control saliente. Siguiendo la primera interpretación, tenemos un lugar marcado inicialmente con un token y conecta mediante un arco a la transición correspondiente al nodo inicial. Este lugar marcado determina que la transición solo se puede disparar una vez, ya que el token se eliminará en el momento del disparo, haciendo que la transición quede inhabilitada a partir de ese momento. Mientras que para la otra interpretación se puede lograr simplemente omitiendo este lugar y el arco; en este caso, la transición podría dispararse en cualquier momento, un número arbitrario de veces.

3.2.2.3 Bifurcaciones

Para poder representar las bifurcaciones procedemos de manera análoga a lo descripto en la sección 3.2.1.3.

3.2.2.4 Divisiones y Uniones

Los nodos de la división tienen un borde de actividad entrante y varios salientes. Cuando el borde entrante está activo, el nodo de la bifurcación activa todas las barreras salientes, de modo que se crean múltiples flujos que se realizarán simultáneamente. Una representación directa de la RdP es una transición única, con arcos entrantes y salientes a los lugares que representan los límites de flujo de control entrantes y salientes, respectivamente. Esto es lo que propone en la sección 3.1 cuando hablamos de concurrencia.

Los nodos de unión marcan un punto de ejecución que solo puede continuar, es decir, activar el borde de salida único, cuando todos los flujos unidos han llegado a él. La representación de la RdP, muy parecida a la del nodo de la horquilla, es una transición única, con arcos entrantes y salientes a los lugares que representan los bordes de flujo de control entrantes y salientes, respectivamente. Esto es lo que propone en la sección 3.1 cuando hablamos de sincronización.

3.2.2.5 Algoritmo de Conversión

Todo lo expuesto anteriormente, se puede resumir en el siguiente pseudocódigo:

Datos de Entrada: *Diagrama de actividad (1)*
Datos de Salida: *RdP Estocástica Etiquetada (2)*

(* Conversión de estados de acción y control de flujo*)
Para cada Acción de (1) hacer
Crear los lugares que representen la ejecución de las mismas
Crear las transiciones que vinculen los anteriores, determinando el tiempo necesario para ejecutarlas.

Para cada Control de flujo de (1) hacer
Crear los lugares necesarios
Crear las transiciones hacia los lugares que representen las actividades

(* Conversión de estado inicial y final*)
Crear un lugar con los tokens necesarios que represente el inicio de la red
Crear las transiciones hacia las actividades correspondientes
Crear un lugar que represente el estado final de la red
Crear las transiciones de las actividades hacia dicho lugar

(*Divisiones y Uniones*)
Para cada División presentes en (1) hacer
Crear la transición que desencadenará las diversas acciones
Crear un lugar por cada lugar
Unir cada uno de los lugares con la transición creada

Para cada Unión presente en (1) hacer
Crear los lugares que representen las acciones
Crear la transición que representa la unión de todas las actividades
Unir cada una de los lugares definidos con la transición creada

Figura 31 Algoritmo de conversión de Diagramas de Actividad a RdP

3.2.3 Diagramas de Secuencia

Los diagramas de secuencia son una herramienta que proporciona UML para representar la dinámica del sistema, desde el punto de vista de cómo se va a comportar un sistema y en qué orden se deben ejecutar los mensajes entre los objetos de tal manera de cumplir con una funcionalidad. Sin embargo, los aspectos dinámicos de los sistemas estudiados en esta tesis no se pueden modelar solamente con estos diagramas, es por esto que para poder obtener la RdP equivalente donde se pueden observar estos aspectos, vamos a considerar los siguientes elementos de los diagramas, con el fin de facilitar la transformación: Mensajes (sincrónicos y asincrónicos), Ciclos de vida, alternativas, ciclos y los puntos de partida y de fin, todos estos elementos se pueden observar en la siguiente imagen.

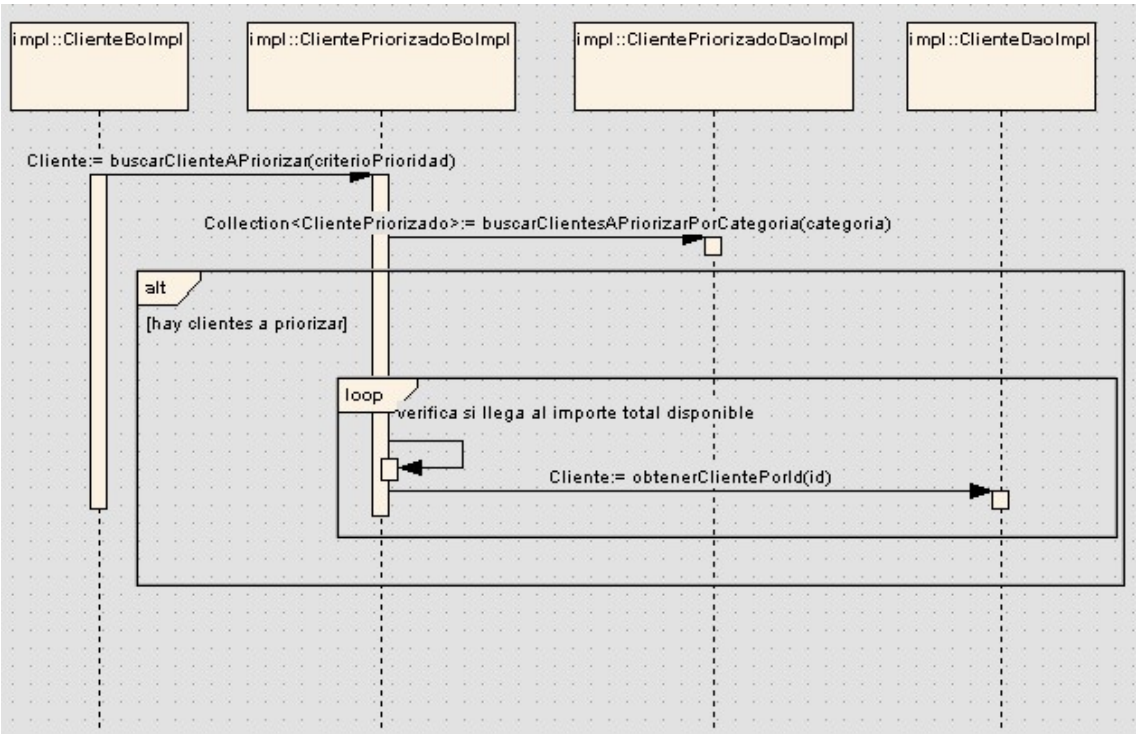


Figura 32 Ejemplo de Diagrama de Secuencia [52]

3.2.3.1 Mensajes

Una de las desventajas del modelo resultante mediante este diagrama, es la falta de notación para representar el tiempo asociado a un mensaje conjuntamente con la posibilidad de determinar si el mismo ocurre de manera sincrónica o asincrónica. Todo esto es posible en las RdP y es por esto que cuando consideramos el mapeo de los mensajes, se plantea el siguiente esquema. Para cada uno de los mensajes, es necesario representar al emisor y el receptor del mismo, por ello, se plantea un lugar que representa el comienzo del proceso de envío de mensaje, seguido por una transición, cuyo tiempo va a depender si lo que consideramos es un mensaje instantáneo o bien este mensaje debe tener un retardo. Esta transición simula el proceso de envío hacia otro lugar de la RdP que representa al receptor del mensaje enviado. Este último realiza las acciones necesarias para poder cumplir con la resolución del mensaje, y utilizando el mismo procedimiento retorna el resultado al objeto emisor, en caso de ser necesario.

3.2.3.2 Línea de Vida

Uno de los elementos más importantes a la hora de poder representar un diagrama de secuencia, es la línea de vida que representa cuando un objeto se encuentra activo para ejecutar el paso de mensajes y poder cumplir con el objetivo para el cual fue creado. Esto en la RdP equivalen queda representado por la sucesión de estados y transiciones que se utilizan en la red para simular cada uno de los diferentes elementos que se describen en esta sección.

3.2.3.3 Inicio y Fin del diagrama

Al igual que en los mapeos anteriores, tanto el inicio como el fin de la RdP equivalente se representa con un lugar, donde el primero posee los tokens que van a circular por el modelo planteado y el segundo va a contener todos los tokens una vez que finalice toda la simulación de la red.

3.2.3.4 Flujo de alternativa

En el modelado de algunos diagramas, es necesario plantear la alternativa en el flujo, dependiendo el cumplimiento de una determinada condición lógica. Para poder modelar esta situación, utilizamos nuevamente lo descripto en la sección 3.1, donde se describe la ejecución concurrente de la red, agregándole en los arcos la probabilidad asociada, en función de la condición booleana establecida.

3.2.3.5 Flujo de ciclo

Como último elemento importante de un diagrama de secuencia para modelar es el flujo de control que corresponde a un ciclo. Para poder modelar este aspecto como una RdP, hay que tener en cuenta la secuencia de lugares y transiciones modeladas a partir de lo expuesto en las secciones anteriores y que se encuentren dentro del cuerpo del ciclo; a esto hay que sumarle la condición de ciclado y fin de la misma. Para esto, volvemos a hacer uso de lo descripto en la sección 3.1, donde se describe la concurrencia, donde en cada una de las alternativas contiene la probabilidad de seguir ciclando y la otra no; y una vez finalizado el cuerpo del ciclo, se realiza la conexión entre la última transición del mismo y el lugar que desencadena el proceso de concurrencia, para poder determinar si es necesario volver a hacer una vuelta del ciclo o no.

3.2.3.6 Algoritmo de Conversión

Lo expresado en esta sección, se puede resumir en el siguiente algoritmo.

Datos de Entrada: *Diagrama de secuencia (1)*
Datos de Salida: *RdP Estocástica Etiquetada (2)*

(* Conversión de mensajes*)
Para cada Mensaje de (1) hacer
Crear los lugares que representen el emisor y receptor
Crear la transición, que parta del emisor, que simule el comienzo del envío del mensaje
Crear el lugar y la transición (con su correspondiente duración) que determinen el envío del mensaje

(* Conversión de estado inicial y final*)
Crear un lugar con los tokens necesarios que represente el inicio de la red
Crear las transiciones hacia las actividades correspondientes
Crear un lugar que represente el estado final de la red
Crear las transiciones de las actividades hacia dicho lugar

(*Flujo de alternativa y Flujo de ciclo*)
Para cada flujo de decisión (d) de (1) hacer
Crear el lugar que representa el comienzo del proceso de alternativa
Para cada una de las transiciones de salida de (d) hacer
Crear el lugar de destino g
Crear la transición del lugar g a s
Establecer el peso de la transición

Para cada flujo de ciclo presente en (1) hacer
Crear las uniones necesarias entre los lugares que conforman las operaciones involucradas dentro del ciclo.

Figura 33 Algoritmo de conversión de Diagramas de Secuencia a RdP

3.2.4 Diagrama de Casos de Uso

El análisis de requisitos juega un papel muy importante en la confiabilidad, el costo y la seguridad de un sistema de software. El enfoque de caso de uso sigue siendo el enfoque dominante durante la obtención de requisitos en la industria. Desafortunadamente, este enfoque tiene varias deficiencias, como la falta de precisión y la dificultad para analizar y validar el comportamiento dinámico de los casos de uso para la concurrencia, la consistencia, entre otros aspectos importantes para los tipos de sistemas bajo estudio en esta tesis. Es por eso que, en esta sección, se presentan los elementos necesarios de los casos de uso y su correspondiente equivalencia en las RdP, que permiten modelar las deficiencias enunciadas anteriormente.

Para el estudio de los casos de uso, es necesario tener en cuenta dos aspectos fundamentales la descripción de un caso de uso, incluida la tarjeta de caso de uso y los escenarios escritos utilizando el lenguaje del escenario y el diagrama de casos de uso, donde de una manera gráfica se representa la conexión entre los diversos casos de uso y los actores involucrados; pero para esta tesis

consideramos que modelando el primer elemento estamos modelado también el segundo, por lo que solo nos centramos en este primero. Se define una subred Petri temporizada y controlada para cada parte de la tarjeta de casos de uso, de los cuales vamos a considerar, la ejecución de un evento simple, la estructura de bifurcación y la estructura de ciclo; y a través de la interconexión de cada subred, se obtiene la red que modela todo el caso de uso.

3.2.4.1 Ejecución de Evento Simple

Para este caso, la traducción es sencilla, lo que se debe tener en cuenta es lo siguiente por cada acción deben definirse el lugar de ingreso a la acción, la transición donde se ejecuta la acción y el lugar que representa la salida de la acción. Cuando definimos la transición, debemos tener en cuenta la temporalización de la misma, en caso de corresponder, incorporando la expresión que permita simular dicho tiempo.

3.2.4.2 Estructura de Bifurcación

Nuevamente, es necesario plantear la alternativa en el flujo, dependiendo el cumplimiento de una determinada condición lógica y para ello, volvemos a utilizar los conceptos desarrollados en la sección 3.1, donde se describe la ejecución concurrente de la red, agregándole en los arcos la probabilidad asociada, en función de la condición booleana establecida.

3.2.4.3 Estructura de ciclado

En esta situación, estamos frente a un problema similar a lo planteado en la sección 3.2.3.5, por lo que la solución del mapeo para la estructura de ciclos, en los casos de uso se resuelve de manera análoga a lo planteado en la sección enunciada anteriormente.

3.2.4.4 Algoritmo de Conversión

Lo expresado en esta sección, se puede resumir en el siguiente algoritmo.

Datos de Entrada: *Diagrama de caso de uso (1)*
Datos de Salida: *RdP Estocástica Etiquetada (2)*

(* Conversión de eventos simples*)
Para cada evento de (1) hacer
Crear los lugares que simulen el ingreso, la ejecución y la salida de la actividad
Crear la transición que una cada uno de los lugares anteriores
Establecer el tiempo de demora para cada una de las transiciones

(*Flujo de alternativa*)
Para cada flujo de decisión (d) de (1) hacer
Crear el lugar que representa el comienzo del proceso de alternativa
Para cada una de las transiciones de salida de (d) hacer
Crear el lugar de destino g
Crear la transición del lugar g a s
Establecer el peso de la transición

Para cada flujo de ciclo presente en *(1)* ***hacer***
Crear las uniones necesarias entre los lugares que conforman las operaciones involucradas dentro del ciclo.

Figura 34 Algoritmo de conversión de Diagramas de Casos de Uso a RdP

Capítulo 4: Aplicación del Enfoque

4.1 Descripción del Sistema bajo estudio

4.1.1 Introducción al Sistema

El sistema planteado para este estudio, corresponde al consumo eléctrico de las viviendas residenciales de la Cuidad de Salta. Se simula el sistema a partir de un modelo discreto, dinámico y estocástico.

A partir de los resultados que se obtienen simulando este sistema, es posible, además, el estudio del consumo eléctrico teniendo en cuenta los siguientes horizontes temporales:

- Horizonte estacional: temporada estival e invernal, no necesariamente discriminada por el calendario, sino por los datos de temperatura obtenidos en el INTA, esto se debe a que el consumo eléctrico no es el mismo en los meses correspondientes a temperatura más elevadas que aquellos relacionados con los meses donde la temperatura es menor.

- Horizonte mensual: esto se debe a que los consumos eléctricos entre los diferentes meses no son lo mismo, ya que analizar el consumo en un mes donde las personas se encuentras de vacaciones no es lo mismo que analizar la demanda de aquellos, donde las personas están todos los días del mes en el domicilio.

- Horizonte semanal: nuevamente se plantea la diferencia en la demanda, entre los días laborales y los fines de semana.

- Horizonte diario: para este caso, teniendo en cuenta los comportamientos de las personas de la Ciudad de Salta, no es lo mismo estudiar el comportamiento durante las horas nocturnas que las horas diarias o bien durante los períodos de tiempo donde los ocupantes se encuentran trabajando que aquellos donde se encuentran en el domicilio.

- Horizonte horario: Teniendo en cuenta los diferentes niveles de abstracción para el problema, este nivel es el más elemental y que da origen y permite estudiar todos los demás, es por esto que este nivel está fuertemente relacionado con la variable de estudio del problema, que es el consumo horario de cada una de las viviendas de la Ciudad de Salta. Sin embargo, para poder calcular el consumo horario, es necesario el

encendido y apagado de artefactos minuto a minuto, para tener una mejor perspectiva de la situación.

Centrándonos en la descripción anterior, podemos decir que nuestra variable de interés es el consumo eléctrico horario. Como lo que se busca es analizar cómo se comporta el sistema a lo largo del tiempo, es que clasificamos el problema como un problema dinámico. Además, este sistema se caracteriza por tener un comportamiento estocástico, esto se debe a que el comportamiento de las personas no se puede definir como un conjunto de reglas determinísticas y es por esto que es necesario contemplar distribuciones de probabilidad asociadas al consumo.

Para poder estudiar este sistema y por la complejidad del objeto de estudio, el tesista se apoya en una metodología que integra lógicas cuantitativas y cualitativas. Entre las primeras, puede decirse que varias decisiones de modelado se basan en numerosas consultas efectuadas a expertos en el área de la energía eléctrica de la universidad y profesionales de la empresa de distribución de energía eléctrica de la provincia de Salta (EDESA), y en encuestas y entrevistas aplicadas a usuarios residenciales. Con estos instrumentos, desde el punto de vista cuantitativo, se elaboraron matrices de datos, que fueron tratados estadísticamente, lo que permitió tener un acercamiento a la realidad investigada y validar el modelo. Además, fueron necesarios estudiar otros factores, que se describen en las secciones siguientes.

4.1.1 Relevamiento de Datos

Para conocer los hábitos de consumo de las distintas viviendas residenciales de la ciudad de Salta se aplicaron encuestas y entrevistas semi-estructuradas a usuarios de distintas zonas geográficas de la ciudad. Se consultaron aspectos relacionados con el consumo energético de cada artefacto presente en el domicilio, discriminándolos de acuerdo a las siguientes categorías:

- Iluminación
- Refrigeración de alimentos
- Calefacción y refrigeración de la vivienda
- Entretenimiento
- Artefactos de cocina
- Lavado y secado de ropa

Se indagó sobre cantidad de artefactos, el tiempo de encendido de los aparatos o bien, la frecuencia semanal de uso.

Se consultó, además:

1. Número de integrantes del grupo familiar, discriminado por edad
2. Períodos de tiempo en el cual no hay personas en el domicilio

3. Períodos de tiempo en el cual hay mínima actividad que requiera energía eléctrica en la vivienda, asumiendo que las personas se encuentran descansando

Estos últimos dos aspectos, están íntimamente relacionados con el comportamiento eléctrico de una vivienda, debido a que, si no existen personas en el domicilio o bien todas duermen, los artefactos no pueden ser prendidos ya que dependen de la acción humana para estar conectados a la red eléctrica.

Para que los resultados obtenidos tengan validez y el error cometido sea el menor posible, es necesario determinar el tamaño de la muestra óptima. Basándose en lo que se propone en [54] y en [55], se utilizó la siguiente expresión para el cálculo del tamaño de la muestra:

Ecuación 2 Calculo del tamaño muestral

$$n = \frac{Z_\alpha^2 N \sigma^2}{e^2(N-1) + Z_\alpha^2 \sigma^2}$$

Donde:

- n: tamaño de la muestra
- N: tamaño de la población
- e: error muestral que se está dispuesto a aceptar
- σ^2: desviación estándar de la población. En la literatura, se determina que, en caso de desconocer dicho valor, se debe utilizar 0.5
- Z_α^2: constante que depende del nivel de confianza que asignemos. Los valores más utilizados en la literatura son:

Tabla 1 Valores más utilizados para Z_α^2 [54]

Z_α^2	**1.15**	**1.28**	**1.44**	**1.65**	**1.96**	**2**	**2.58**
Nivel de Confianza	75%	80%	85%	90%	95%	95,5%	99%

Para asegurar que la muestra integre la mayor variedad de tipos de viviendas de la ciudad, fue necesario considerar la clasificación de las viviendas que propone EDESA, de acuerdo a su cuadro tarifario, para la realización de las encuestas y las entrevistas. Se consideró el total de viviendas provistas por EDESA teniendo en cuenta una confianza del 95%.

Tabla 2 Categorización de viviendas (EDESA) y Cantidad de viviendas a encuestar

Categoría	Consumo Máximo Mensual (kWh)	Total de Viviendas	Sub-Total a Encuestar
R1	Hasta 192	160570	391
R2	Entre 192 y 500	116964	391
R3	Entre 500 y 700	13382	381
R4	Entre 700 y 1400	7765	373
R5	Más de 1400	1319	302
		Total a encuestar:	**1838**

Otro insumo para el modelo, además de los datos obtenidos por las encuestas, fue contar con datos climáticos, los cuales fueron provistos por el Instituto Nacional de Tecnología Agropecuaria (INTA). Estos, se utilizaron para desarrollar y validar los modelos que simulan la temperatura horaria y la humedad relativa, los cuales se describen con mayor detalle en la sección siguiente.

4.1.2 Simulación de Datos Climáticos

Las variables relacionadas con el clima, juegan un rol fundamental a la hora de pronosticar el consumo energético, ya que estas variables definen la zona de confort térmico, que determina si una persona desea o no prender y/o apagar un artefacto de calefacción o refrigeración. Por esta razón, fue necesario introducir al modelo los procedimientos necesarios para poder estimar tanto la temperatura como la humedad relativa.

En [56] se propone un procedimiento para simular la humedad relativa de una ciudad, considerando la misma en períodos horarios. Para ello, se tiene en cuenta, la temperatura mínima y media hasta el período anterior al que se desea simular.

Dicho procedimiento, se basa en la aplicación de la siguiente ecuación:

Ecuación 3 Calculo de la presión atmosférica

$$HR = 100\frac{e_s(T_n)}{e_s(T_m)}$$

Donde $e_s(T_n)$ corresponde a la presión de saturación (hPa), determinada por la temperatura mínima del día. Mientras que $e_s(T_m)$ corresponde también a la presión de saturación, pero calculada a partir de la temperatura horaria. La expresión que permite calcular estas presiones es:

Ecuación 4 Calculo de la presión de saturación

$$e_s(T) = 6{,}11 * e^{25{,}22\left(1-\frac{213{,}16}{T}\right)} * \left(\frac{213{,}16}{T}\right)^{5{,}31}$$

Posteriormente, se procedió a validar este procedimiento, a partir de los datos climáticos provistos por el INTA. Para ello se calculó el error cuadrático medio (RMSE) que arrojó un valor de 2.66%. Por lo tanto, el procedimiento es considerado válido desde el punto de vista estadístico, para simular la humedad relativa.

En [58] se proponen diversos métodos para la simulación de la temperatura horaria de una ciudad. Para ello, es necesario contar con la temperatura mínima y máxima del día anterior y la hora solar aparente del período a simular.

Ecuación 5 Calculo de la temperatura ambiente

$$T_{ab} = T_1 + T_2 \cos\left((14 - AST)\frac{\pi}{12}\right)$$

Donde

- T_{ab} corresponde a la temperatura ambiente
- AST corresponde a la hora solar aparente
- T_1 y T_2 corresponden a la salida de las siguientes ecuaciones:

Ecuación 6 Cálculo del valor auxiliar

$$T_1 = \left(\frac{T_{max} + T_{min}}{2}\right)$$

Ecuación 7 Calculo de valor auxiliar

$$T_2 = \left(\frac{T_{max} - T_{min}}{2}\right)$$

Este procedimiento, permite calcular únicamente para un día en particular; entonces se modificó con el fin de simular las temperaturas horarias de todo un año.

Ecuación 8 Modificación para el cálculo diario de la temperatura

$$T_{max}(i+1) = T_{max}(i) + value1$$

Ecuación 9 Modificación para el cálculo diario de la temperatura

$$T_{min}(i+1) = T_{min}(i) + value2$$

Donde, a las temperaturas máximas y mínimas a considerar para el cálculo del día siguiente, se les adicionan value1 y value2, que se obtienen del análisis de las distribuciones que siguen las variaciones mensuales de las temperaturas. Este estudio, es realizado a partir de los datos provistos por el INTA.

Para la validación de este modelo, se calculó RSME, el cual arrojó un valor de 4.23%. Por lo tanto, este modelo puede ser considerado válido desde el punto de vista estadístico, para simular la temperatura horaria durante todo un año.

4.1.3 Salida y Puesta del sol

Otro aspecto importante incorporado al modelo, son los procedimientos que calculan la salida y la puesta del sol, ya que estos valores están íntimamente relacionados con el intervalo de tiempo de encendido o apagado de artefactos, se utilizaron las siguientes fórmulas [58]:

Ecuación 10 Cálculo de la salida del sol

$$h_{ss} = 12 - \frac{\omega_s}{15}$$

Ecuación 11 Cálculo de la puesta del sol

$$h_{ps} = 12 + \frac{\omega_s}{15}$$

Donde ω_s corresponde al ángulo horario a la puesta y salida del sol y se calcula como:

Ecuación 12 Cálculo del ángulo horario

$$\omega_s = \text{acos}(-\tan\delta * \tan\phi)$$

De esta última ecuación se define

- ϕ como la latitud del lugar
- δ como la declinación que se obtiene de la siguiente fórmula, que posee como dato de entrada el día del año, expresado en días julianos

Ecuación 13 Cálculo de la declinación

$$\delta = 23{,}45\, sen\left(360\frac{284+n}{365}\right) \qquad (\delta\ en\ grados)$$

Los resultados obtenidos por las ecuaciones 10 y 11 están expresados en función de la Hora Solar Real (HSR), pero lo que necesitamos para trabajar es la Hora Solar Aparente (HSA) y para ello se utiliza la siguiente ecuación:

Ecuación 14 Cálculo de la hora real en función de la hora solar aparente

$$HSA = HSR - 4\ (L_s - L_E) - E_t$$

Donde,

- L_s corresponde al meridiano de referencia
- L_E corresponde al meridiano del lugar
- E_t corresponde a la ecuación del tiempo que se obtiene a partir de la siguiente fórmula

Ecuación 15 Cálculo de la ecuación del tiempo

$$E_t = 229{,}2 * (0{,}000075 + 0{,}001868\cos B - 0{,}032077\ sen\ B - 0{,}014615\cos 2B - 0{,}04089\ sen\ 2B)$$

Por último, el valor de B corresponde al ángulo diario que se obtiene a partir de:

Ecuación 16 Cálculo del ángulo diario

$$B = (n-1)\frac{360}{365}$$

Donde n corresponde al día del año expresado en tiempo juliano.

4.1.4 Encendido y Apagado de Artefactos

Para simular el encendido y apagado de los artefactos, fue necesario clasificar los artefactos en las siguientes categorías:

1. Artefactos que se encuentran conectados a la red eléctrica durante todo el día, como ser teléfonos inalámbricos y módems. Para ellos, el tiempo de encendido son las 24 horas del día.

2. Artefactos cuyo funcionamiento depende de la cantidad de luz natural que exista en una determinada hora del día. Son artefactos que se caracterizan porque su funcionamiento tiene una estrecha relación entre la hora actual y las horas de salida y puesta del sol. Esto se debe a que, en períodos de escasa luz natural, se tiende a tener encendido por más tiempo el artefacto, que en períodos donde hay abundancia luz natural. Para el cálculo de los tiempos de encendido/apagado, se utilizan los resultados de las ecuaciones 9, 10 y 13 de la sección 4.1.3. Dentro de esta categoría están los artefactos relacionados con la iluminación de la vivienda.

3. Artefactos relacionados con la climatización de las viviendas. Para determinar el encendido y apagado de estos artefactos, es necesario simular la temperatura y humedad relativa horaria, con las ecuaciones desarrolladas en la sección 4.1.2. De los valores obtenidos y la definición de la región denominada zona de confort[2], se definió el procedimiento para el encendido/apagado de los artefactos de esta categoría. El algoritmo analiza si los valores de temperatura y humedad relativa se encuentran dentro de la zona de confort[3], de ser así el artefacto debía ser apagado, en caso de estar encendido. En caso contrario, el modelo aplica el encendido del artefacto de acuerdo a la relación entre las variables y la región de confort, ya que, ante temperaturas inferiores a las de la zona de confort, deben encenderse los artefactos relacionados con la calefacción;

[2] El confort térmico se define en la Norma ISO7730 como "Condición del ambiente en la que se expresa la satisfacción con el ambiente térmico", en función de la temperatura y la humedad relativa, como variables más importantes a destacar de la definición.

[3] Para la ciudad de Salta, la zona de confort queda definida entre los valores de temperatura 20 y 35 grados Celsius y la humedad relativa entre el 30% y el 60% (*Cartas Psicométricas de la Provincia de Salta*)

mientras que, si las temperaturas se encuentran por encima de la zona de confort, se deben encender los artefactos relacionados a la refrigeración de las viviendas.

4. Artefactos relacionados con el lavado y secado de ropa. Se consideró la cantidad de uso por día y el ciclo de lavado más utilizado para determinar el tiempo de funcionamiento del aparato. Para artefactos relacionados con el secado, el algoritmo posee una condición extra de control, que considera el encendido sólo si previamente se registra el uso de una lavadora de ropa.

5. Artefactos que poseen batería interna que le otorga autonomía una vez desconectados de la red eléctrica. Para estos artefactos, el encendido depende exclusivamente de que la batería no se haya agotado o esté a punto de hacerlo. Se controló que el artefacto no se encuentre dentro de su período de autonomía para ser nuevamente conectado a la red eléctrica. Como ejemplo de esta categoría tenemos, teléfonos celulares y computadoras portátiles.

6. Para los restantes artefactos, se generaron al azar el tiempo de funcionamiento.

Cabe destacar, que cada uno de estos procedimientos se ejecuta bajo el doble control: que la vivienda cuente con el artefacto a encender y que en el domicilio se encuentren personas.

4.2 Aplicación del enfoque desarrollado

Está claro de las definiciones de la estructura de red, la representación de estado y de las reglas de disparo, que existe un principio de localidad en los estados y las acciones. La localidad de la representación del estado y de las acciones es otra característica crucial de las RdP, ya que la misma permite el modelado de conductas distribuidas complejas. Su importancia para la síntesis de modelos reside en el hecho que las redes pueden ser localmente modificadas, o bien, refinadas sin ninguna alteración del resto del modelo. También es posible que diferentes redes, modelando diferentes partes de un sistema, puedan estar compuestas por algunos nodos compartidos (representando actividades y estados comunes). En otras palabras, las redes pueden estar sintetizadas usando las aproximaciones top-down y bottom-up.

Como sabemos, en términos informales, una RdP es un grafo dirigido, bipartito, en el cual las clases de vértices que puede haber son lugares (P, por el nombre en inglés Places) y transiciones (T por el nombre en inglés Transition). El marcado de una red asigna a cada lugar un entero no negativo, que representa la cantidad de *tokens* que existen en cada lugar. La ejecución de una RdP hace que la

marcación cambie y de esta manera los *tokens* viajen a través de los arcos. El flujo de *tokens* está regulado por las transiciones.

En el sistema bajo estudio se pueden distinguir dos grandes aspectos, por un lado, la generación de cada una de las viviendas de la Ciudad, teniendo en cuenta la cantidad de cada categoría y los artefactos con los que cuenta. Por otro lado, está el proceso nudo del problema que es la simulación del apagado y encendidos de los artefactos de cada hogar, evaluando a cada minuto cada situación de los artefactos y calculando los valores correspondientes, para obtener la variable de interés, que recordemos es el consumo eléctrico horario de cada vivienda.

A la hora de realizar el análisis y diseño del mismo, utilizando los modelos y las gráficas que propone UML, podemos observar que este lenguaje no cuenta con las herramientas, procesos o artefactos necesarios para modelar el paso del tiempo, aspecto fundamental para poder calcular la variable de interés, porque es necesario evaluar el sistema hora a hora, minuto a minuto para determinar su comportamiento y determinar los artefactos encendidos o apagados de tal manera que ayuden a calcular la demanda eléctrica horaria.

Es por esto que, a los fines de modelar el sistema, se desea utilizar una RdP Temporalizada, usando semántica de competencia para resolver conflictos. Se incluyen transiciones inmediatas, además de las transiciones temporalizadas. Aquellas transiciones no inmediatas toman un cierto tiempo estocástico antes de disparar, el denominado retardo. Además, se utiliza la política de memoria de habilitación de reajuste (los retrasos en las transiciones se muestran tras cada disparo), por otro lado, el dominio a utilizar es de tiempo discreto, ya que se analiza el sistema a intervalos fijos de tiempo.

A pesar de todo lo descripto anteriormente, ***para poder determinar el tiempo de encendido de cada uno de los artefactos de los hogares, es necesario plantear un escenario estocástico, ya que el comportamiento de los encendidos sigue reglas estocásticas, ya que proviene del comportamiento humano, el cuál es muy azaroso***. Entonces es por esto que se plantean el uso de las RdP Estocásticas Generalizadas Etiquetadas.

Los lugares representan recursos, estados locales y variables del sistema. Las transiciones representan acciones, eventos o decisiones. Además, las transiciones pueden ser inmediatas o temporalizadas. Las primeras, representan acciones o eventos que son irrelevantes bajo el punto de vista del rendimiento del sistema. Mientras que las segundas, tienen prioridad sobre las inmediatas y pueden tener un retardo de tiempo determinístico o probabilístico antes de disparar.

Habiendo definido el contexto de modelado, como punto de partida, se representa el modelo del encendido y apagado de artefactos sumamente abstracto. El cual se irá refinando hasta el nivel de detalle deseado, obteniendo

así el modelo final. Este modelo proviene del diagrama de estado que se muestra en la siguiente figura.

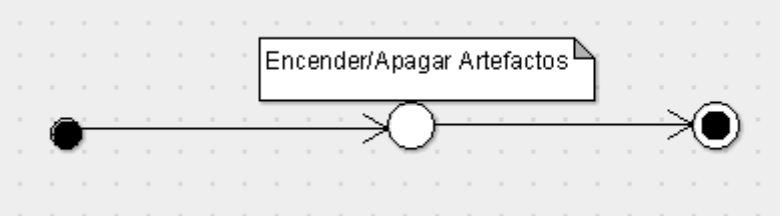


Figura 35 Diagrama de Estados correspondiente al proceso de calcular el consumo horario

El proceso de encendido y apagado de los artefactos, desde un punto de vista muy abstracto, podría considerarse como un sistema constituido por un gran proceso cuya finalidad es encender y apagar los artefactos hogareños a lo largo de cada una de las horas del día y producir como salida los consumos correspondientes a cada una de ellas. Así, en la figura 36 se muestra el mapeo del modelo de UML descripto anteriormente a la correspondiente RdP.

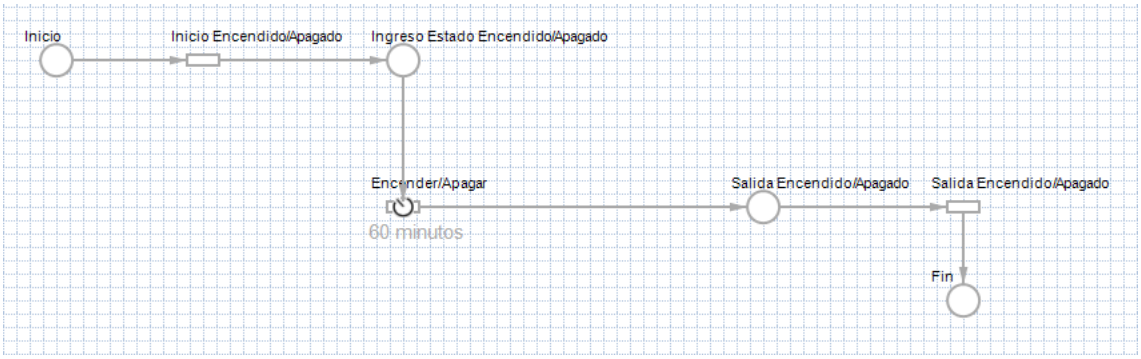


Figura 36 RdP equivalente al diagrama de estados de la figura 35

El modelo simplista en consideración, está compuesto por una transición temporalizada, cuatro lugares que corresponden con los procesos asociados al inicio y fin del proceso, como así también a las entradas a las diferentes acciones a ejecutar dentro del modelo. Además, este modelo cuenta con dos transiciones inmediatas y una transición temporalizada que es la que se encarga de ejecutar todo el proceso de encendido y apagado de los artefactos y el posterior cálculo de la demanda eléctrica horaria. Esta transición depende de la ejecución de las acciones relacionadas con el proceso de simulación de los horizontes temporales superiores, teniendo en cuenta lo explicado en la sección 4.1.1.

El lugar *Ingreso Estado Encendido/Apagado* modela el procesamiento de todos los artefactos con los que puede llegar a contar una determinada vivienda de la ciudad, de este modo queda evidenciado el nivel de abstracción del modelo. Mientras que la transición *Encender/Apagar,* como su nombre lo indica, modela la actividad de simular el prendido y/o apagado de los artefactos de la vivienda. Su disparo queda determinado por el proceso que se dispara cuando se desea calcular el consumo eléctrico horario para cada una de las viviendas de la ciudad.

Finalmente, la transición asociada al *Salida Encendido/Apagado* se encarga de realizar los cálculos para obtener una observación de la variable aleatoria de nuestro interés.

Como puede observarse este modelo abstracto no hace distinción entre los encendidos y apagados de los diferentes tipos de artefactos que se describieron anteriormente. Tampoco nos brinda información sobre los procesos estocásticos llevados a cabo para poder determinar si un artefacto se debe encender o no, y en caso de corresponder, el tiempo de encendido del mismo. Entonces, el siguiente paso, consiste en refinar el modelo intentando dar más detalle.

El siguiente paso consiste en refinar la transición *Encender/Apagar* considerando como está conformado el núcleo del problema. En la figura 37 se muestra el refinamiento. Para referenciar, las transiciones temporalizadas son aquellas que cuentan con el dibujo de un reloj y debajo de la transición cuenta con la expresión matemática que permite calcular el tiempo asociado o bien está explicitado cuál es dicho tiempo; mientras que las transiciones que no son temporalizadas no cuentan con lo descripto anteriormente. Además, se considera que todos los lugares tienen capacidad uno, salvo que se especifique lo contrario.

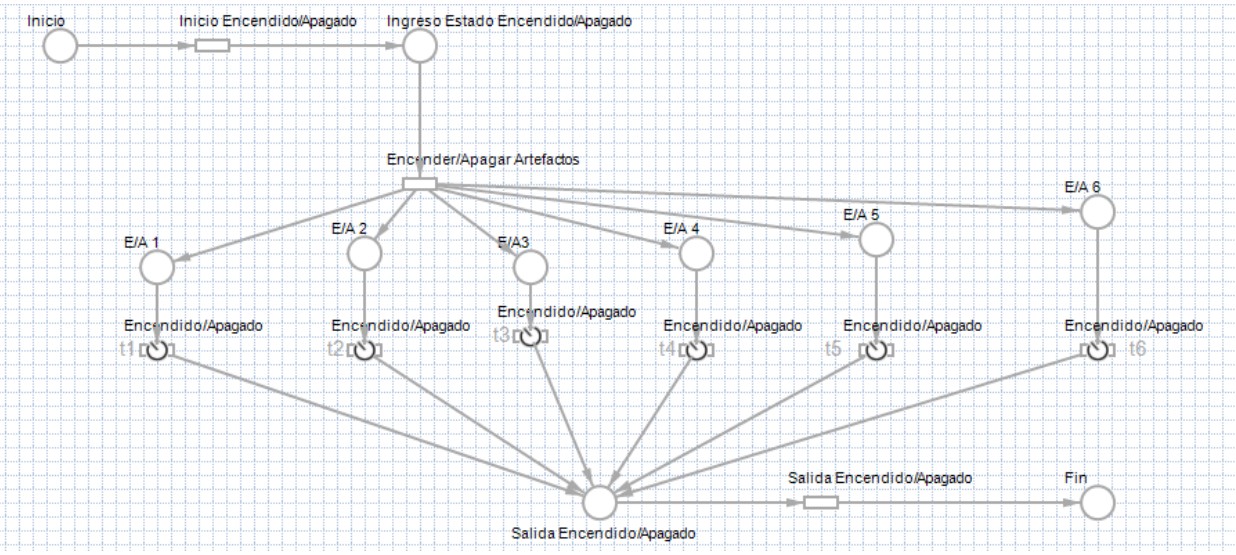


Figura 37 Refinamiento del modelo planteado en la figura 36

Para facilitar la notación del modelo, se procedió a simplificar los nombres de las transiciones, a continuación, se describen cada una de las notaciones usadas:

- E/A 1: Corresponde al encendido y/o apagado de los artefactos que van a estar encendidos durante toda la hora que se está simulando.

- E/A 2: Corresponde al encendido y/o apagado de los artefactos que están ligados con la presencia de luz natural.

- E/A 3: Corresponde al encendido y/o apagado de los artefactos que están relacionados con la climatización de una vivienda.

- E/A 4: Corresponde al encendido y/o apagado de los artefactos que están relacionados con el lavado y secado de ropa.

- E/A 5: Corresponde al encendido y/o apagado de los artefactos que se caracterizan por tener una batería interna.

- E/A 6: Corresponde al encendido y/o apagado de los artefactos del resto de los artefactos.

Por otro lado, también fue necesario simplificar la notación de los tiempos asociados a cada transición, es por eso que las variables t_i corresponde al tiempo que se encontrará encendido un artefacto.

En este último modelo, falta realizar la distinción para determinar el tiempo de encendido dependiendo de la hora del día que estemos analizando, ya que no es lo mismo el tiempo de encendido de las luces durante la mañana que durante la noche, por ejemplo; o no es lo mismo el tiempo que una persona usa un determinado artefacto durante las horas del día que durante la noche que es cuando se encuentra durmiendo, es por esto que la refinación al modelo planteado en la figura 37, con estas incorporaciones, queda de representado de la siguiente manera.

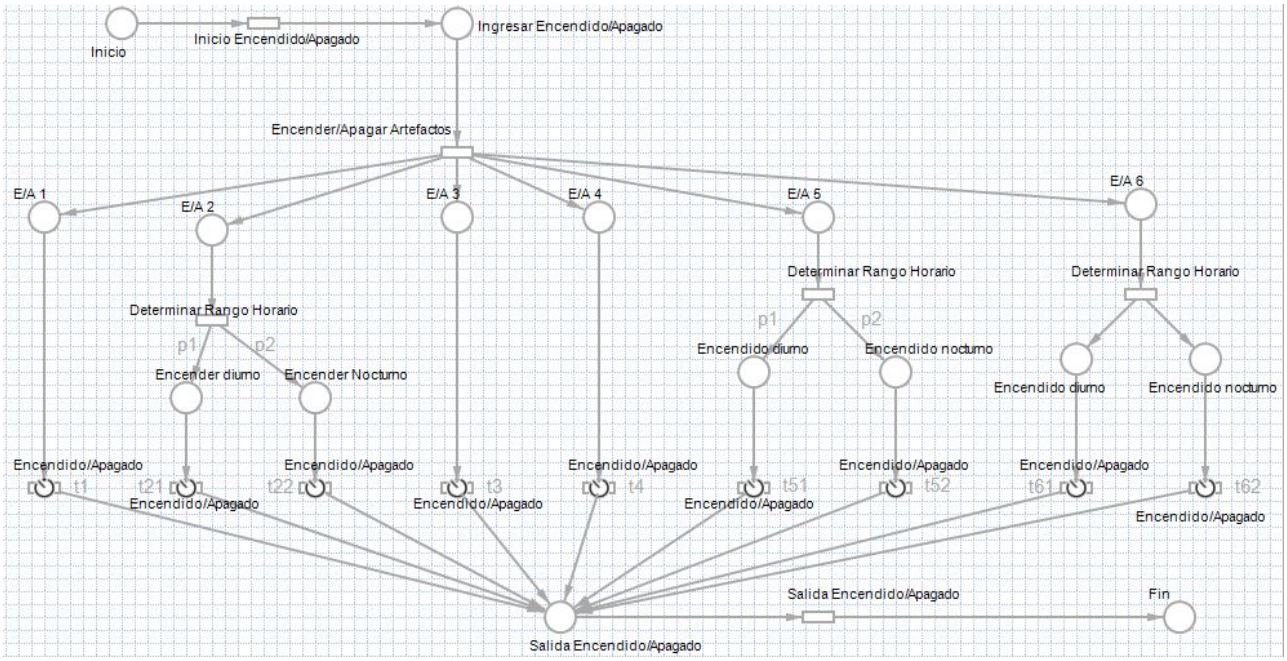


Figura 38 Refinamiento del modelo planteado en la figura 37

Por último, falta analizar el comportamiento de los encendidos/apagados de los artefactos, teniendo en cuenta la presencia de personas en el domicilio, para ello refinamos el modelo propuesto en la figura 38, obteniendo lo siguiente.

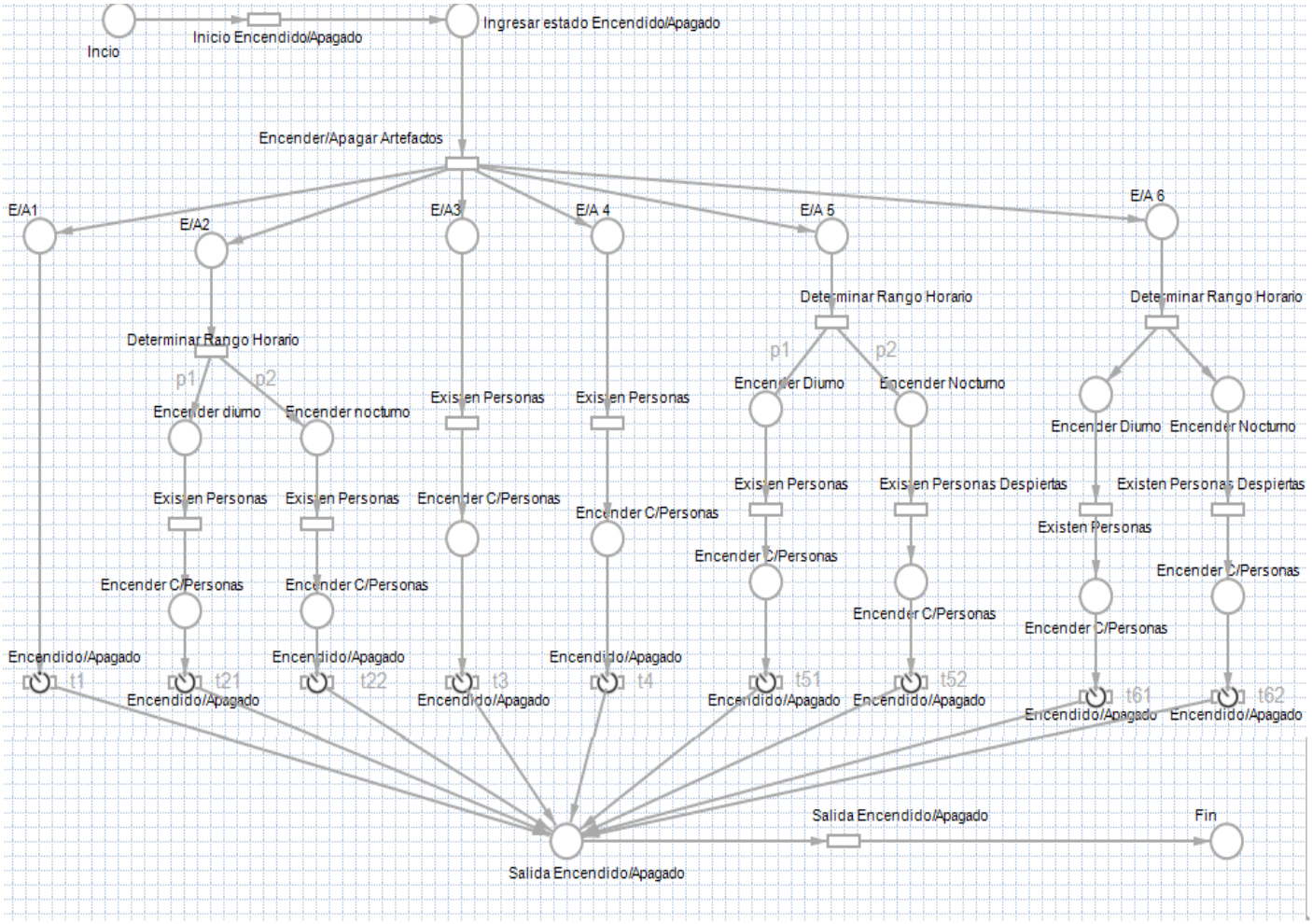


Figura 39 Refinamiento final modelo planteado

Con este último modelo, se considera que se representa con un suficiente nivel de abstracción, la realidad estudiada y permite el cálculo de la demanda eléctrica horaria para una casa en particular, este modelo se replica para todas y cada una de las viviendas de la ciudad de Salta.

Cabe notar que el modelo presentado no es más que una serie de elecciones y lugares de contabilización que actúan como variables de sistema. Para modelar estas elecciones se utilizó una semántica de competencia, es decir, transiciones temporalizadas para que una vez habilitadas compitan entre ellas, para ver cuál dispara primero. La meta es modelar una cuantía a través de dicha competencia.

Los disparos se producen siguiendo una cuantía, dicha función de probabilidad se obtiene del análisis de los datos estadísticos provenientes de las encuestas de comportamiento eléctrico realizadas a los residentes de la ciudad de Salta. Es posible simular la cuantía con transiciones temporalizadas y puestas en conflicto

entre sí, cuyos retrasos se calculan siguiendo una distribución uniforme; o sea, siguiendo la semántica de competencia.

Para la construcción de la red, se utilizó el software HPSim [59], ya que se trata de una herramienta de apoyo al diseño de las RdP. Al diseñar una RdP Estocástica, se necesita indicar el resto de los parámetros que componen la definición, más concretamente los retrasos de las transiciones temporalizadas; esto es, la distribución de probabilidad con la que se obtendrán los diferentes retrasos de las transiciones mencionadas. Esta información, se obtiene del análisis de los resultados de las encuestas y se resumen en las siguientes tablas.

Tabla 3 Retardo de Transiciones Temporalizadas

Retardo	Distribución	Retraso Inicial	Retraso Final
t_1	Ninguna	60 minutos	
t_{21}	Uniforme	2 minutos	12 minutos
t_{22}	Uniforme	35 minutos	50 minutos
t_3	Uniforme	20 minutos	30 minutos
t_4	Uniforme	15 minutos	35 minutos
t_{51}	Uniforme	45 minutos	50 minutos
t_{52}	Uniforme	50 minutos	55 minutos
t_{61}	Uniforme	15 minutos	30 minutos
t_{62}	Uniforme	5 minutos	10 minutos

Además, de la información de los retrasos es necesario especificar las cuantías asociadas a las variables aleatorias que determinan la presencia o no de personas en la vivienda y la que determine si existen personas durmiendo o no en una determinada hora del día.

Tabla 4 Cuantía de la Variable Aleatoria que determina presencia de personas en el domicilio

Referencia	Rango Horario	Probabilidad
Existen Personas	(8 – 13 horas)	0.3
No existen Personas	(8 – 13 horas)	0.7
Existen Personas	(16 – 21 horas)	0.56
No existen Personas	(16 – 21 horas)	0.44
Existen Personas	Resto del día	0.8
No existen personas	Resto del día	0.2

Tabla 5 Cuantía de la Variable Aleatoria que determina la existencia de personas despiertas

Referencia	Rango Horario	Probabilidad
Existen personas despiertas	(23 – 8 horas)	0.13
No existen personas despiertas	(23 – 8 horas)	0.87
Existen Personas despiertas	Resto del día	0.9
No existen personas despiertas	Resto del día	0.1

Capítulo 5: Análisis de resultados y conclusiones

En este capítulo se presenta los resultados de la ejecución del modelo planteado con el enfoque propuesto, validando los resultados a partir de los datos provistos por la empresa distribuidora de energía eléctrica de la Provincia de Salta (EDESa). Posteriormente, se presentan las conclusiones del trabajo realizado en esta tesis.

5.1 Análisis de Resultados

Los datos corresponden al consumo residencial, de todas las viviendas de la ciudad de Salta durante tres años. Los resultados obtenidos se detallan en la tabla siguiente.

Tabla 6 Comparación de Resultados

	Consumo 2014 (GWh)	Consumo 2015 (GWh)	Consumo 2016 (GWh)	Consumo Simulado (GWh)
Enero	74.89	76.01	84.57	83.04
Febrero	59.61	67.00	75.44	77.29
Marzo	61.08	70.47	73.02	73.18
Abril	58.11	63.88	69.16	74.45
Mayo	62.61	67.47	80.99	81.77
Junio	68.25	71.88	88.15	84.58
Julio	72.18	78.95	83.10	81.97
Agosto	65.78	71.21	72.67	74.90
Septiembre	61.36	65.25	70.03	73.30
Octubre	68.37	70.08	72.45	75.42
Noviembre	68.24	72.57	72.30	77.10
Diciembre	72.11	81.73	83.32	81.10

El error cuadrático medio, RMSE, entre los datos simulados y los datos del año 2014 es 12.77; el RMSE entre los datos simulados y los datos del año 2015 es 8.05; y el valor del RMSE entre los datos simulados y los datos del año 2016 es 2.9. Mientras que valor promedio del RMSE para los 3 años es de 7.9.

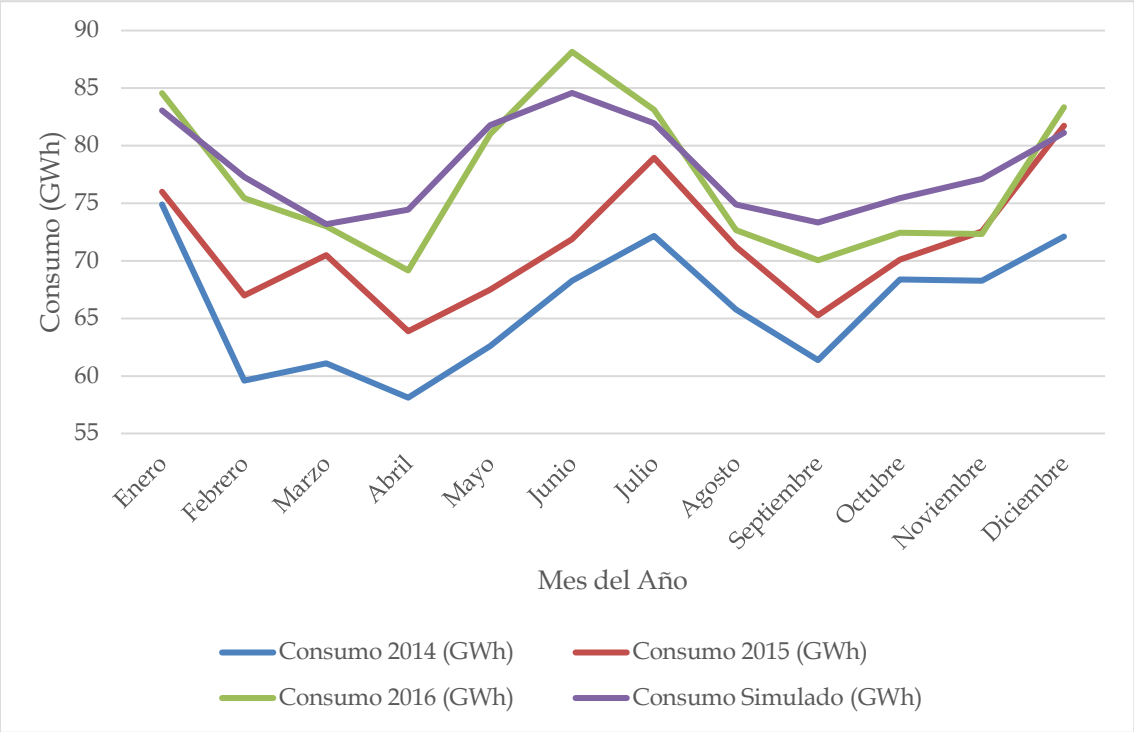


Figura 40 Validación de datos de salida de la Simulación

En la figura 40, se muestra la comparación entre los valores reales de los años 2014 a 2016 y los valores simulados. La ejecución se realizó en una computadora de 8 GB de memoria RAM con procesador Intel Core 7. Respecto a los tiempos de cómputo, en la siguiente tabla se muestran los tiempos, discriminados por tiempo de generación promedio del tipo de vivienda y el tiempo promedio de simulación de esa vivienda.

Tabla 7 Tiempos de Generación y Simulación

Categoría de la vivienda	Tiempo Generación (seg)	Tiempo de Simulación (seg)
R1	499.03	49.25
R2	151.41	36.24
R3	177.22	36.29
R4	154.60	36.15
R5	508.23	36.45

5.2 Conclusiones

En este trabajo de tesis se presentó una propuesta de enfoque para analizar y diseñar los sistemas dinámicos, estocásticos y discretos, a través de las herramientas y artefactos que proponen UML con una extensión de las RdP Estocásticas Generalizadas Etiquetadas, como medio para complementar los primeros y así poder cubrir con las características típicas de los sistemas bajo estudio, sobre todo las características asociadas con el tiempo.

Uno de los requisitos principales en el modelado y análisis de los sistemas estudiados en este trabajo, es que los modelos de diseño deben ser inequívocos, precisos y verificables. De acuerdo con este requisito, los expertos han sugerido varios métodos que combinan el método orientado a objetos con métodos formales.

A cada uno de los elementos de los modelos propuestos por UML, se establecen las reglas de mapeos para obtener la RdP equivalente, con estos elementos se modelan las características necesarias para poder representar los modelos estudiados. En este trabajo se implementaron dichas reglas de mapeo y se presentan en el capítulo 3, obteniendo como resultado, la posibilidad de modelar todos los sistemas dinámicos, estocásticos y discretos, a través de dichas reglas.

Las conclusiones remiten al marco teórico, en cuyo contexto se interpretan los resultados obtenidos. A partir de la literatura vigente, las contribuciones de este trabajo de tesis se pueden caracterizar de la siguiente manera. La primera, consiste en el aislamiento de las deficiencias o problemas de las metodologías formales de análisis y diseños de los sistemas presentados en este trabajo; deficiencias que se salvan utilizando los formalismos de las RdP y esto se debe gracias a la versatilidad y adaptación a los cambios que poseen la metodología y herramientas abarcadas en esta tesis. Aunque se sugirieron varias RdP de alto nivel con los conceptos de objetos con una idea clara en preocupaciones específicas, no soportaban completamente las características suficientes que se necesitan en el modelado de sistemas con conceptos orientados a objetos. Para resolver este problema, se utilizaron las RdP estocásticas generalizadas etiquetadas, que admiten la mayoría de las características de los conceptos orientados a objetos con una semántica clara, conjuntamente con las características de los sistemas estudiados.

Además, se describen los métodos de modelado y análisis a partir de un desarrollo incremental e iterativo. Esto se puede lograr desde la base de objetos encapsulados y modularizados, con información resumida, descomposición y enfoque de refinamiento para el modelado de sistemas y análisis de accesibilidad incremental. Los resultados, que permiten afirmar que el enfoque propuesto es beneficioso y útil para poder trabajar los sistemas mencionados, se centran en los siguientes:

- Una investigación y problemas abiertos en el modelado de software basado en escenarios, es resolver la diferencia de nivel de abstracción para cada escenario. Cuando cada escenario se analiza y modela con métodos formales, y el nivel de abstracción de cada escenario difiere, la integración de cada escenario representado formalmente es problemática. En estudios relacionados con modelos basados en escenarios, este problema se resuelve asumiendo que cada escenario está representado por el mismo nivel de abstracción. Sin embargo, esta suposición no es realista y no elimina los obstáculos para el modelado de software utilizando

escenarios. El enfoque propuesto es adecuado para resolver este problema porque es posible representar naturalmente la abstracción.

- La reutilización ha sido reconocida como una tecnología clave que puede generar importantes ganancias de productividad en el desarrollo de software. La reutilización en los requisitos de software también es otra esfera de la que se pueden esperar grandes beneficios. Las redes modeladas utilizadas pueden considerarse un componente reutilizable, ya que cada modelo de red fue modularizado y encapsulado con una interfaz formal. El mérito clave de reutilizar los modelos es que no se necesitan mecanismos adicionales para respaldar la reutilización.

Finalmente, a partir de los resultados obtenidos en la sección 5.1, donde se presenta el análisis estadístico de los datos arrojados por la simulación, obtenida por aplicación del enfoque desarrollado, podemos concluir que el trabajo propuesto en esta tesis constituye un gran aporte a la bibliografía existente sobre la temática abordada en esta tesis; ya que el modelo planteado refleja la realidad de los hábitos de consumo de la Ciudad de Salta y esto se debe a contar con buenas herramientas para realizar el análisis del sistema objeto.

Capítulo 6: Bibliografía Referenciada y Consultada

[1] : Pezz, G. D. and M. (2004). Petri Nets and Software Engineering. Lecture Notes in Computer Science, Volume 309, pp 439-466. Retrieved from http://www.inf.uni-konstanz.de/soft/teaching/ws13/seminar/9.pdf.

[2] : Murata, T. (1989). Petri Nets: Properties, Analysis and Applications. Proceedings of the IEEE. https://doi.org/10.1109/5.24143.

[3] : Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. Educational and Psychological Measurement, 20(1), 37–46. http://doi.org/10.1177/001316446002000104.

[4] : Silva, José & Almança, Eston & Santos, Dos. (2004). Applying Petri Nets to Requirements Validation. ABCM Symposium. Series In Mechatronics. 1. 10.1016/S1474-6670(17)36190-6.

[5] : Zhao, J., & Duan, Z. (2009). Verification of use case with Petri nets in requirement analysis. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). https://doi.org/10.1007/978-3-642-02457-3_3.

[6] : Andrade, E., Maciel, P., Callou, G., Nogueira, B., & Araújo, C. (2009). Mapping UML sequence diagram to time petri net for requirement validation of embedded real-time systems with energy constraints. Proceedings of the 2009 ACM Symposium on Applied Computing - SAC '09. https://doi.org/10.1145/1529282.1529364.

[7] : Yang, N., Yu, H., Sun, H., & Qian, Z. (2010). Mapping UML activity diagrams to analyzable Petri net models. In Proceedings - International Conference on Quality Software. https://doi.org/10.1109/QSIC.2010.26.

[8] : Basile, F., Chiacchio, P., & Del Grosso, D. (2009). A two-stage modelling architecture for distributed control of real-time industrial systems: Application of UML and Petri Net. Computer Standards and Interfaces. https://doi.org/10.1016/j.csi.2008.03.021.

[9] : Staines, T. S. (2008). Intuitive mapping of UML 2 activity diagrams into fundamental modeling concept Petri net diagrams and colored Petri nets. In Proceedings - Fifteenth IEEE International Conference and Workshops on the Engineering of Computer-Based Systems, ECBS 2008. https://doi.org/10.1109/ECBS.2008.12

[10] : López-Grao, J. P., Merseguer, J., & Campos, J. (2004). From UML activity diagrams to Stochastic Petri nets. In Proceedings of the fourth international workshop on Software and performance - WOSP '04. https://doi.org/10.1145/974044.974048.

[11] : Yang, N., Yu, H., Sun, H., & Qian, Z. (2010). Mapping UML activity diagrams to analyzable Petri net models. In Proceedings - International Conference on Quality Software. https://doi.org/10.1109/QSIC.2010.26.

[12] : (Oxímetro de Pulso) (https://es.wikipedia.org/wiki/Ox%C3%ADmetro_de_pulso) (30/12/2018).

[13] : Hei, X., Chang, L., Ma, W., Gao, J., & Xie, G. (2011). Automatic transformation from UML statechart to Petri nets for safety analysis and verification. In ICQR2MSE 2011 - Proceedings of 2011 International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering. https://doi.org/10.1109/ICQR2MSE.2011.5976760.

[14] : Ameedeen, M. A., Bordbar, B., & Anane, R. (2011). Model interoperability via Model Driven Development. Journal of Computer and System Sciences. https://doi.org/10.1016/j.jcss.2010.01.011.

[15] : Faria, J. P., & Paiva, A. C. R. (2016). A toolset for conformance testing against UML sequence diagrams based on event-driven colored Petri nets. International Journal on Software Tools for Technology Transfer. https://doi.org/10.1007/s10009-014-0354-x.

[16] : Yang, N., Yu, H., Sun, H., & Qian, Z. (2010). Modeling UML sequence diagrams using extended Petri nets. In 2010 International Conference on Information Science and Applications, ICISA 2010. https://doi.org/10.1109/ICISA.2010.5480384.

[17] : Jørgensen, J. B., Tjell, S., & Fernandes, J. M. (2009). Formal requirements modelling with executable use cases and coloured Petri nets. In Innovations in Systems and Software Engineering. https://doi.org/10.1007/s11334-009-0075-6.

[18] : Krause, J., Hintze, E., Magnus, S., & Diedrich, C. (2012). Model based specification, verification, and test generation for a safety fieldbus profile. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). https://doi.org/10.1007/978-3-642-33678-2_8.

[19] : Distefano, S., Scarpa, M., & Puliafito, A. (2011). From UML to Petri nets: The PCM-based methodology. IEEE Transactions on Software Engineering. https://doi.org/10.1109/TSE.2010.10.

[20] : Emadi, S., & Shams, F. (2009). Transformation of usecase and sequence diagrams to petri nets. In 2009 Second ISECS International Colloquium on Computing, Communication, Control, and Management, CCCM 2009. https://doi.org/10.1109/CCCM.2009.5267604.

[21] : Danjiang, Z. & Huobin, T. & Shuzhen, Y. (2018). Petri Nets-based method to elicit component-interaction related safety requirements in safety-critical systems. Computers & Electrical Engineering. 71. 162-172. https://doi.org/10.1016/j.compeleceng.2018.07.019.

[22] : Á. Rocha et al. (eds.), New Perspectives in Information Systems and Technologies, Volume 2, Advances in Intelligent Systems and Computing 276, https://doi.org/10.1007/978-3-319-05948-8_13.

[23] : Noguera, M., Hurtado, M. V., Rodríguez, M. L., Chung, L., & Garrido, J. L. (2010). Ontology-driven analysis of UML-based collaborative processes using OWL-DL and CPN. Science of Computer Programming. https://doi.org/10.1016/j.scico.2009.05.002.

[24] : Liao, H., Wang, Y., Cho, H. K., Stanley, J., Kelly, T., Lafortune, S., … Reveliotis, S. (2013). Concurrency bugs in multithreaded software: Modeling and analysis using Petri nets. Discrete Event Dynamic Systems: Theory and Applications. https://doi.org/10.1007/s10626-012-0139-x.

[25] : Kelly, T., Wang, Y., Lafortune, S., & Mahlke, S. (2009). Eliminating concurrency bugs with control engineering. Computer. https://doi.org/10.1109/MC.2009.391.

[26] : Grobelna, I., Wisniewska, M., Wiśniewski, R., Grobelny, M., & Mroz, P. (2014). Decomposition, validation and documentation of control process specification in form of a Petri net. In Proceedings - 2014 7th International Conference on Human System Interactions, HSI 2014. https://doi.org/10.1109/HSI.2014.6860481.

[27] : Wu, N. Q., Zhou, M. C., & Li, Z. W. (2015). Short-Term Scheduling of Crude-Oil Operations: Enhancement of Crude-Oil Operations Scheduling Using a Petri Net-Based Control-Theoretic Approach. IEEE Robotics and Automation Magazine. https://doi.org/10.1109/MRA.2015.2415047.

[28] : Alfonso, E., Xie, X., Augusto, V., & Garraud, O. (2012). Modeling and simulation of blood collection systems. Health Care Management Science. https://doi.org/10.1007/s10729-011-9181-8.

[29] : Windhager, L., & Zimmer, R. (2008). Intuitive modeling of dynamic systems with Petri nets and fuzzy logic. Proc German Conf Bioinf.

[30] : Sadiq, A., Ahmad, F., Khan, S. A., Valverde, J. C., Naz, T., & Anwar, M. W. (2014). Modeling and analysis of departure routine in air traffic control based on Petri nets. Neural Computing and Applications. https://doi.org/10.1007/s00521-014-1590-4.

[31] : Mashkov, V., Barilla, J., & Simr, P. (2013). Applying petri nets to modeling of many-core processor self-testing when tests are performed randomly. Journal of Electronic Testing: Theory and Applications (JETTA). https://doi.org/10.1007/s10836-012-5346-8.

[32] : Yasuda, G. (2012). Implementation of real-time distributed control for discrete event robotic systems using Petri nets. Artificial Life and Robotics. https://doi.org/10.1007/s10015-011-0984-y.

[33] : Vanit-Anunchai, S. (2018). Modelling and simulating a Thai railway signalling system using Coloured Petri Nets. International Journal on Software Tools for Technology Transfer. https://doi.org/10.1007/s10009-018-0482-9.

[34] : Lozada, M., & Velasco, J. M. (2010). Modelado dinámico basado en Redes de Petri para el modelo de integración empresarial "Actor de Empresa." Scientia et Technica.

[35] : Córdova, F. M., & Cifuentes, F. (2016). Stochastic Dynamic Simulation Model Applied to Public Lawyers Using Petri Nets. In Procedia Computer Science. https://doi.org/10.1016/j.procs.2016.07.136.

[36] : Cabac, L., Haustermann, M., & Mosteller, D. (2018). Software development with Petri nets and agents: Approach, frameworks and tool set. Science of Computer Programming. https://doi.org/10.1016/j.scico.2017.12.003.

[37] : Khan, S. A., Zafar, N. A., Ahmad, F., & Islam, S. (2014). Extending Petri net to reduce control strategies of railway interlocking system. Applied Mathematical Modelling. https://doi.org/10.1016/j.apm.2013.06.002.

[38] : Brant-Ribeiro, T., Araújo, R. D., Mendonça, I. E., Soares, M. S., & Cattelan, R. G. (2017). Interactive web interfaces modeling, simulation and analysis using Colored Petri Nets. Software and Systems Modeling. https://doi.org/10.1007/s10270-017-0593-x.

[39] : Offutt, J., & Thummala, S. (2018). Testing concurrent user behavior of synchronous web applications with Petri nets. Software and Systems Modeling. https://doi.org/10.1007/s10270-018-0655-8.

[40] : Bernardi, S., Campos, J., & Merseguer, J. (2011). Timing-failure risk assessment of UML design using time petri net bound techniques. IEEE Transactions on Industrial Informatics. https://doi.org/10.1109/TII.2010.2098415.

[41] : Xu, D. (2011). A tool for automated test code generation from high-level Petri nets. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). https://doi.org/10.1007/978-3-642-21834-7_17.

[42] : Davidrajuh, R., & Lin, B. (2011). Exploring airport traffic capability using Petri net based model. Expert Systems with Applications. https://doi.org/10.1016/j.eswa.2011.02.134.

[43] : Nabih, A. K., Gomaa, M. M., Osman, H. S., & Aly, G. M. (2011). Modeling, simulation, and control of smart homes using petri nets. International Journal of Smart Home.

[44] : Silva, D., Fernandes, J. M., & Belo, O. (2013). Assisting data warehousing populating processes design through modelling using coloured petri nets. In SIMULTECH 2013 - Proceedings of the 3rd International Conference on Simulation and Modeling Methodologies, Technologies and Applications.

[45] : Du, Y., Qi, L., & Zhou, M. (2014). Analysis and Application of Logical Petri Nets to E-Commerce Systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems. https://* doi: 10.1109/TSMC.2013.2277696.

[46] : Chen, Y., Li, Z., & Zhou, M. C. (2014). Optimal supervisory control of flexible manufacturing systems by petri nets: A set classification approach. IEEE Transactions on Automation Science and Engineering. https://doi.org/10.1109/TASE.2013.2241762.

[47] : Boukredera, D., Maamri, R., & Aknine, S. (2016). Stochastic Petri net-based modeling and formal analysis of fault tolerant Contract Net Protocol. Web Intelligence. https://doi.org/10.3233/WEB-160342.

[48] : Wu, D., & Schnieder, E. (2018). Scenario-based system design with colored Petri nets: an application to train control systems. Software and Systems Modeling. https://doi.org/10.1007/s10270-016-0517-1.

[49] : Liu, F., Bla¨tke, M. A., Heiner, M., & Yang, M. (2014). Modelling and simulating reaction-diffusion systems using coloured Petri nets. Computers in Biology and Medicine. https://doi.org/10.1016/j.compbiomed.2014.07.004.

[50] Zimmermann, A. (2008). Stochastic Discrete Event Systems. Stochastic Discrete Event Systems. https://doi.org/10.1007/978-3-540-74173-2.

[51] Booch, G., Rumbaugh, J., & Jacobson, I. (1999). El Lenguaje Unificado de Modelado. Elements. https://doi.org/1852 – 4516.

[52] (Diagrama de Secuencia – Dos Ideas)(https://dosideas.com/wiki/Diagrama_de_secuencia) (07/04/2019).

[53] Zhao, J., & Duan, Z. (2009). Verification of use case with Petri nets in requirement analysis. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). https://doi.org/10.1007/978-3-642-02457-3_3.

[54] García-García, J. A., Reding-Bernal, A., & López-Alvarenga, J. C. (2013). Cálculo del tamaño de la muestraeninvestigacióneneducaciónmédica. InvestigaciónEnEducaciónMédica, 2(8), 217–224. https://doi.org/10.1016/S2007-5057(13)72715-7.

[55] Valdivieso, C., Valdivieso, R., &Valdivieso, O. (2011).Determinación del tamaño muestral mediante el uso de árboles de decisión. UPB-Investigación&Desarrollo, 11(2000), 148–176. https://doi.org/10.1007/s13398-014-0173-7.2.

[56] Meira, G. R., Padaratz, I. J., Alonso, C., & Andrade, C. (2003).Efecto de la distancia al mar en la agresividadporcloruroseneestructuras de hormigónen la costa brasileña. Materiales de Construcción, 53(271–272), 179–188. Retrieved from http://materconstrucc.revistas.csic.es/index.php/materconstrucc/article/view/302/347.

[57] Chabane, F., Moummi, N., Brima, A., &Moummi, A. (2016). Prediction of the theoretical and semi-empirical model of ambient temperature. Frontiers in Energy, 10(3), 268–276. https://doi.org/10.1007/s11708-016-0413-y.

[58] J.A. Duffie, W.A. Beckman, Solar Engenieering of therma processes, Editorial Wiley & Sons, INC.

[59] Página oficial HPSim (www.winpesim.de)