ESTUDIO COMPARATIVO DE LAS HERRAMIENTAS CASE: STARUML, POSEIDON FOR UML Y ENTERPRISE ARCHITECT, PARA EL MODELAMIENTO DE DIAGRAMAS UML

DANIEL LÓPEZ ORTEGA
JESSICA ANDREA SANTA VILLA

UNIVERSIDAD TECNOLÓGICA DE PEREIRA
FACULTAD DE INGENIERÍAS
PROGRAMA INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
PEREIRA
2012

ESTUDIO COMPARATIVO DE LAS HERRAMIENTAS CASE: STARUML, POSEIDON FOR UML Y ENTERPRISE ARCHITECT, PARA EL MODELAMIENTO DE DIAGRAMAS UML

DANIEL LÓPEZ ORTEGA
JESSICA ANDREA SANTA VILLA

PROYECTO DE INVESTIGACIÓN COMO REQUISITO PARA OPTAR AL TÍTULO
DE INGENIERO DE SISTEMAS Y COMPUTACIÓN

DIRECTOR DE PROYECTO
LUIS EDUARDO MUÑOZ

UNIVERSIDAD TECNOLÓGICA DE PEREIRA
FACULTAD DE INGENIERÍAS
PROGRAMA INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
PEREIRA
2012

Nota de aceptación:

_____

_____

_____

_____

_____

_____

Firma del Presidente del Jurado

_____

Firma del Jurado

Febrero, 15 de 2012

AGRADECIMIENTOS

JESSICA ANDREA SANTA VILLA

Primero que todo a Dios por permitir este nuevo logro en mi vida, y a mi padre aunque ya no está acá en la tierra siempre estuvo y estará en mi corazón, sin sus enseñanzas y valores jamás hubiese logrado la culminación de esta etapa de vida.

A mi madre, por su paciencia porque supo entender mis momentos de estrés y ocupación para no compartir tiempo de calidad con ella, por sus palabras de ánimo cada que desfallecía y me dejaba derrotar por las barreras que se me presentaron a lo largo de estos años de carrera. Gracias mamita por creer en mí, porque cuando muchos dudaron solo tu seguiste firme con tu amor y compresión.

A mi abuelita, por sus dulces palabras y sonrisas, porque siempre estuvo a mi lado dándome apoyo, porque hoy quiero devolverle todo el orgullo que ella siente por mí, por ser mi segunda madre y brindarme sus valores y formación para ser la mujer y profesional en la que me convertiré.

A mi hermana y mi tía por su apoyo incondicional, porque siempre han creído en mí.

A Emmanuele porque desde que llego a nuestras vidas es el angelito que me da ilusión e ilumina mi camino para cada día ser mejor y generar un ejemplo a seguir en su vida.

A mi tía Sor Melida Santa, a mi tía Rubí santa y sus hijos, por su apoyo y preocupación, por no olvidar que existía y en todo momento darme un lugar en sus corazones y su familia, gracias y mil bendiciones.
A mis tíos Jhon Jairo, Juan Darío y Martha Cardona y familia en general por todo el apoyo, porque de una u otra manera impulsaron mi sueño de ser una ingeniera, su apoyo es este camino se convirtieron en más fuerza para lograr este nuevo triunfo en mi vida, el cual es solo el inicio de muchos éxitos venideros.

A mis compañeros y amigos por toda la paciencia y buenos recuerdos que aportan a mi vida, a lo largo de estos años de universidad han sido muchas las personas que pasaron por mi vida y de una u otra manera han dejado un impacto significativo.

A mi compañero de proyecto por su compromiso y consejos, por creer en este proyecto, por creer en mí como su compañera, por confiar que podíamos sacar esta idea adelante. A nuestro director de proyecto por sus aportes y consejos.

A todos los docentes que aportaron a mi formación como profesional, que han dejado parte de sus conocimientos y experiencias plasmadas en estos años de formación.

Mil bendiciones, Jessica Andrea Santa Villa

DANIEL LOPEZ ORTEGA

Una etapa más de mi vida superada, una meta más alcanzada, una enseñanza más aprendida, una experiencia más obtenida, un sinnúmero de amistades forjadas y una verdad más comprendida. Todo esto gracias a las personas que me acompañaron durante estos años de estudio, en los cuales pude explorar un universo de conocimientos que ahora hacen parte de mí y de la persona que hoy soy.

Pero no solo de experiencias se compone la sabiduría, pues todo depende de cómo decidamos aplicarlas. Es por eso que doy gracias a mis Padres por enseñarme desde pequeño a ver el mundo con ambición, con sed de conocimiento, no solo para obtener sabiduría, sino para honrar la verdad que de ella se deriva. Ellos pusieron su confianza en mí desde que tengo memoria, me guiaron por los caminos de la esperanza, con grandes y pequeños sacrificios, corriendo riesgos y aprendiendo de mis errores. Mamá, tú que siempre te esforzaste por enseñarme desde pequeño, incluso cuando no conocías la respuesta de algo la buscabas y si no eras tú, hallabas a alguien que resolviera mis necesidades de aprender. Gracias por el amor, el cariño, el afecto y el apoyo que me diste y me das día tras días. Papá, tú que, sin importar que llegaras de una larga jornada laboral, compartes las noches con nosotros tus hijos y nos compartes las experiencias que la vida te ofrece. Gracias por el enorme esfuerzo que realizaron sin esperar nada a cambio, con la expectativa de que aquel ser que trajeron al mundo se convirtiese en un hombre de bien, humildemente les demuestro mi agradecimiento no solo con palabras sino con hechos mostrándoles que si puedo ser esa persona que me enseñaron ser. Ustedes Mamá y Papá fueron quienes hicieron de mí un ser humano. Mi hermano, con quien he compartido algo más que el mismo techo, más que una gran familia, más que un apellido; gracias por todo el apoyo que me brindas día a día, por las enseñanzas que me das, por ser un hombre sabio, un hermano bueno, y un amigo sincero.

A mis abuelos que siempre confiaron en mí como ser humano y estuvieron al tanto de todos mis progresos, de mis dificultades, de mis alegrías incluso de mis tristezas. A mi gran familia, que uno a uno me aportan a mi desarrollo personal y profesional.

Gracias a los profesores, que en algún punto de mi Carrera compartieron un espacio conmigo. De muchos de ustedes aprendí lo que puede inspirar un buen líder, que no es más que la grandeza. Afronté miedos y sufrí cambios, pues para madurar primero hay que cambiar. Trasnochos, madrugadas, desvelos y demás, me enseñaron que para alcanzar mis metas debía esforzarme, por pequeño que fuera mi objetivo. Aprendí que realizar sacrificios, incluso pequeños, dan esperanza y la esperanza te da experiencia; la experiencia construye confianza, y la confianza es la que te lleva a cumplir tus sueños.

Gracias a mis compañeros, amigos y demás conocidos que ocupan un gran espacio en mi corazón. Con ellos viví experiencias que fortalecieron nuestras relaciones y que me hicieron crecer como persona.

A todos ellos quiero decirles gracias, porque siempre serán parte de mi, de la persona que hoy soy.

A Jessica Santa, que me brindó la oportunidad de hacer parte de esta iniciativa y de aportar un pequeño granito de conocimiento a este universo del que hacemos parte.

A todos aquellos que no nombro, pero que llevo en mi corazón y en mi mente presentes, porque para mí todos son amigos y de todos he aprendido.

Gracias de corazón, Daniel López Ortega.

CONTENIDO

Pág.

LISTA DE TABLAS

Pág.

LISTA DE FIGURAS

Pág.

INTRODUCIÓN

Desde los inicios de la computación y concretamente del desarrollo de software se han realizado avances a gran velocidad. A partir de la década de los 90 se adaptó un modelo de desarrollo diferente al de cascada, que era el que se utilizaba hasta entonces, en este caso se buscaba tener en cuenta la participación de los usuarios, porque el concepto funcional que tiene el usuario es diferente del punto de vista del desarrollador, este nuevo modelo pretendía crear una estructura ligada al problema y a los usuarios quienes son la conexión directa con la necesidad, hacer la cadena entre el diario vivir y la sistematización, que es la revolución de la era. El equipo encargado de llevar a cabo el diseño y desarrollo, tiene como objetivo final cumplir con las necesidades reales del usuario; Porque lo que la gran mayoría desconoce es lo que hay detrás de este tipo de soluciones. Si se da una mirada más profunda, se cae en cuenta de que el desarrollo de Software depende de varios procesos dentro de los cuales se encuentra el análisis de este, también denominado ingeniería de requisitos[1].

El desarrollo de software pretende cumplir un objetivo, dentro del cual debe seguir un ciclo para ser alcanzado, en este ciclo se tienen una serie de procesos, entre los cuales se destaca la ingeniería o análisis de software, es allí donde se debe pensar en el uso de herramientas CASE, puesto que hacía falta un mecanismo o herramienta de soporte para el proceso de documentación y es así como surgen las herramientas CASE del inglés Computer Aided Software Engineering, Ingeniería asistida por computadoras.

[1] Kenneth E. Kendall, Julie E. Kendall, Antonio Núñez Ramos. Análisis y diseño de sistemas. Sexta Edición. México 2005. p.14 - 15

Que no son más que un respaldo a este campo, con el fin de aumentar la productividad en el desarrollo del Software, la productividad es la relación entre el tiempo usado y el resultado obtenido y va de la mano directamente con la mejora continua de la calidad. Por consiguiente, de lo dicho anteriormente tener en cuenta dentro de los recursos de desarrollo las herramientas CASE brindará un apoyo integral en el mejoramiento de la calidad.

Dado que en la Web fácilmente se encuentran algunos estudios que informan sobre Desarrollo de software, Análisis y Diseño de Software, Ingeniería de Software incluyendo el uso de herramientas CASE que permiten el modelamiento en UML y facilitando tanto el proceso de asignación de tareas dentro del grupo de desarrollo como el uso y la aceptación del sistema a desarrollar por parte del usuario final. Gracias a esto se observa que no existe un respaldo que permitiese verificar las características y ventajas de las herramientas que existen para el desarrollo de estas tareas, la cual se enfoque en las tres herramientas como tal, si se encuentran muchos documentos pero no alcanzan a suplir las necesidades del entorno. Es por eso que surge esta iniciativa, por medio de 3 herramientas (POSEIDON FOR UML, STAR UML y ENTERPRISE ARCHITECT); las cuales son las más usadas y se encuentran al alcance del entorno social académico y profesional del programa y la región para el diseño de software, se realiza un análisis para conocer como dichas herramientas satisfacen las características que ofrecen las compañías a cargo de su desarrollo, distribución y mantenimiento.

Por tal razón cuando los desarrolladores de software se enfrentan la vida laboral, detectan las falencias profesionales en cuanto a los estándares y el uso, de qué tan importante es la decisión de usar o no algún tipo de herramienta. En este enfoque se encuentra un apoyo para decidir qué tipo de herramienta usar y cuál será la más adecuada teniendo claro el alcance y requerimientos del desarrollo, para cumplir exitosamente con la culminación del proyecto.

Como ya se ha mencionado queda claro que se deben usar herramientas CASE, la meta a la cual pretende llegar este documento simplemente es brindar una guía basada en criterios y métricas de calidad, que sea un apoyo a la persona que requiera elegir la herramienta. Esta evaluación será enfocada a tres herramientas pero las métricas pueden extenderse a las miles herramientas existentes en el mercado o la web.

El contenido encontrado acá es generado por la ardua investigación y basado en los conocimientos adquiridos, haciendo un análisis de las experiencias vividas durante el proceso educativo y de generación de proyectos de desarrollo, se eligen tres herramientas CASE para modelamiento UML: **STARUML, POSEIDON FOR UML Y ENTERPRISE ARCHITECT,** consideradas altamente importantes en este campo y en el entorno social, por ser "amigables" con el desarrollador.

STARUML y POSEIDON FOR UML son de libre instalación y uso, pero POSEIDON FOR UML tiene algunas restricciones de las opciones que permiten no son libre, tienen un costo, por el contrario STARUML en su totalidad es libre. ENTERPRISE ARCHITECT es una herramienta de las más completas, pero no es libre, y es un poco más compleja de usar para usuarios comunes.

La evaluación de cada herramienta arrojará las conclusiones de ciertos criterios de evaluación que han sido escogidos basados en la norma de estándares de calidad ISO 9126.

"El sentido común es el menos común de los sentidos"

Voltaire

FORMULACION DEL PROBLEMA

Los proyectos de software presentan necesidades explicitas de herramientas CASE, para la asistencia en sus proyectos, puesto que los ciclos de vida de los proyectos de software, necesitan control constante de las tareas que se realicen en estos mismos. Por lo tanto se hace necesario que el equipo de desarrollo tenga en cuenta cual herramienta CASE es más adecuada para su proyecto a desarrollar.

Parte integral de las herramientas CASE son los diagramas UML, pues por medio de estos se permite verificar acordemente las tareas que el sistema realiza, la manera en como las realiza y además facilita la documentación en todas las fases del proyecto. Generando así un ambiente en el cual tanto el equipo de trabajo como el cliente permanezcan enterados de cómo es que el sistema se desarrolla con respecto al tiempo.

Teniendo en cuenta lo anterior y dada la importancia de estas herramientas, se hace crítica la decisión de cuál de estas herramientas se debe usar. Por tal motivo es necesario ser muy precisos y precavidos al momento de elegir una herramienta para modelar, ya que dependiendo de los requerimientos y los recursos con que se cuente, varía la decisión.

La decisión que se tome con respecto a las HERRAMIENTAS CASE dentro del equipo de desarrollo de Software influye directamente en los resultados, pero a su vez afecta el proceso de desarrollo ya que una herramienta que no se adapte a las necesidades del proyecto para el cual esta siento usada, desestabiliza tanto en tiempo como en dinero la integridad del sistema que se esté desarrollando, y por consecuencia al cliente, quien es el actor más importante en un proyecto.

Este documento debe servir de apoyo al desarrollo de software tanto educativo como empresarial, donde se evalué las herramientas CASE basados en normas internacionales de calidad, como apoyo a los procesos y productos de software, puesto que en un mundo competitivo en el área de los desarrollos es primordial tener en cuenta los principios de calidad.

OBJETIVO GENERAL

Realizar una evaluación comparativa de las herramientas CASE Star UML, Poseidon For Uml y Enterprise Architect, por medio de una adaptación del estándar internacional para la evaluación de la calidad del Software ISO 9126 .

OBJETIVOS ESPECÍFICOS

- Destacar la importancia de usar herramientas CASE para modelamiento UML en el ciclo de vida del desarrollo de software.

- Realizar un reconocimiento de cada una de las herramientas CASE: Star UML, Poseidon For UML y Enterprise Architect, que serán evaluadas durante el transcurso del proyecto.

- Crear un caso de estudio que se implementará en las tres herramientas CASE para el modelamiento UML: Star UML, Poseidon y Enterprise Architec para poder evaluarlas.

- Construir y aplicar el modelo de evaluación con el cual serán evaluadas las tres herramientas, este modelo será basado en el estándar internacional para la evaluación de la calidad del Software ISO 9126.

- Basados en el modelo de evaluación que se creara en el transcurso del desarrollo del proyecto, llegar a una tabla de conclusión porcentual que contendrá el nivel de cumplimiento de cada herramienta con respecto a los criterios de evaluación de la ISO 9126.

HIPÓTESIS

Se pretende comprobar o desarrollar en este proyecto el siguiente interrogante:

¿Es posible crear un documento sólido, conciso y acertado, que sirva al proceso de ingeniería de software, para elegir una herramienta CASE de apoyo pertinente a los proyecto de desarrollo de Software?

Para probar la hipótesis es necesario crear un mecanismo de evaluación que califique cada una de las herramientas respecto a algunas pautas de calidad. Las cuales se apoyaran en las normas establecidas en la ISO9000, específicamente en el estándar internacional para la evaluación de la calidad del Software ISO 9126.

1. IMPORTANCIA DE USAR HERRAMIENTAS CASE

Se consideró importante destacar porqué se deben usar estas herramientas, por tal razón se levanta una serie de incógnitas las cuales serán solucionadas y así poder argumentar concisamente porqué este proyecto y el uso de estas herramientas.

1.1 ¿QUÉ SON LAS HERRAMIENTAS CASE?

Las herramientas de ingeniería de software asistida por computadora (CASE), son aplicaciones computacionales en conjunto que soportan y ayudan al proceso de análisis y desarrollo de Software. Las cuales sirven a los analistas de sistemas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida del desarrollo del Software.

1.2 ¿CUÁNDO USAR HERRAMIENTAS CASE?

En todas las etapas del desarrollo de software. Cuando se inicia un proyecto de software integral bajo las normas de calidad se debe pensar en usar herramientas CASE, ya que estas apoyan al desarrollador y al cliente desde la fase inicial hasta la fase final.

Un ejemplo claro, de cuando usarlas, es en la fase de análisis, los programadores o diseñadores en esta fase realizan la adquisición de requisitos funcionales y no funcionales, para tal fin se recurre a los diagramas de caso de uso, y para el diseño de estos, allí es cuando se deben usar las herramientas CASE.

1.3 ¿POR QUÉ Y PARA QUE USAR HERRAMIENTAS CASE?

Permiten normalizar y automatizar parte de los proceso de desarrollo de software, para lograr mayor calidad en el producto final y así poder obtener una alta portabilidad y migración, tanto del lenguaje como del motor de bases de datos[2].

Justificando lo anterior se discuten algunas razones del porqué y para que usar estas herramientas, se consideran altamente importantes por las siguientes razones:

- Mejora el producto final.
- Facilita el progreso de los procesos.
- Reduce los tiempos.
- Asegura la coherencia y consistencia en los procedimientos.
- Captura los datos del sistema.

1.4 COMPONENTES

De una forma esquemática se puede decir que una herramienta CASE se compone de los siguientes elementos:

- Repositorios: Lugar en el que se almacenan aquellos elementos que hacen parte de la herramienta o que fueron definidos por esta. Su administración se encuentra a cargo de algún Sistema de Gestión de Base de Datos (SGBD).

[2] http://books.google.com.co/books?id=Z0fUgdnVHdgC&pg=PA327&dq=que+son+herramientas+CASE&hl=es&sa=X&ei=lBweT8jzONCFsAKdhoHMDg&ved=0CDgQ6AEwAg#v=onepage&q=que%20son%20herramientas%20CASE&f=false.

- Metamodelo: Definición de los métodos y las técnicas que hacen parte de la herramienta, no siempre es visible.

- Carga o descarga de datos: Capacidad de generar rutinas, diagramas de base de datos, entre otros, para alimentar diferentes sistemas; estos es lo que caracteriza la comunicación con otras herramientas .Facilidad de uso los diferentes elementos de la herramienta.

- Comprobación de errores: Brinda la capacidad de analizar los niveles de consistencia exactitud e integridad de los esquemas que se pueden generar por medio de la herramienta.

- Interfaz de usuario: Es aquel entorno que facilita al usuario por medio de objetos representados gráficamente y editables, generar y modificar los diferentes diagramas, esquema, matrices y demás, por medio de periféricos de entrada; mouse, digitalizer tablet, touch screen etc.

1.5 ¿ESTRUCTURA GENERAL DE LAS HERRAMIENTAS CASE?

La estructura general de las herramientas CASE se divide en tres partes:

- **CASE de alto nivel**: Herramientas que apoyan en la fase de planeación, análisis y diseño de sistemas durante el ciclo de vida del desarrollo, es decir en las fases finales.

- **CASE de bajo nivel:** Al igual que las herramientas de alto nivel, apoyan al ciclo de vida del desarrollo pero estas ayudan en las fases de diseño detallado de sistemas, la implantación de sistemas y el soporte de sistemas, es decir, en las fases intermedias.

- **CASE cruzado de ciclo de vida**: Herramientas que apoyan la estimación y las actividades de gestión de proyectos, que son aquellas tareas que tienen lugar durante todo el ciclo de vida.

1.6 EVOLUCIÓN DE LAS HERRAMIENTAS CASE

A inicios de los 80's:

- Ayuda en la documentación por computadora.
- Diagramación asistida por computadora.
- Herramientas de análisis y diseño.

A mediados de los 80's:

- Diseño automático de análisis y pruebas.
- Repositorios automáticos de información de sistemas.

Al final de los 80's:

- Generación automática de código desde especificaciones de diseño.

A inicios de los 90's:

- Metodología Inteligente.
- Interface de Usuario reusable como una metodología de desarrollo[3].

En la Actualidad:

- Intercambio de Diagramas.
- Superestructura que facilita el uso de los diagramas más comunes.
- Infraestructura mejorada.

[3] Herramienta CASE http://www.inei.gob.pe/biblioineipub/bancopub/Inf/Lib5103/Libro.pdf

2. HERRAMIENTAS CASE USADAS EN EL ESTUDIO

En este capítulo se realiza una investigación acerca de cada una de las tres herramientas, las cuales gracias a la investigación arrojan un conjunto de características que se mencionan posteriormente, con las cuales se observa el producto desde el punto de vista de los desarrolladores, pero no implica que realmente ocurra o sea totalmente influyente en la evaluación definitiva de este proyecto, puesto que basados en ciertos criterios de evaluación que se evidenciaran en el trascurso del proyecto se genera un criterio propio.

2.1 POSEIDON FOR UML

2.1.1 Introducción. Es una de las herramientas CASE orientada a objetos, que cuenta con un amplio y completo grupo de diagramas para el modelamiento UML, tales como: Diagramas de estado, diagramas de clases, diagrama de paquetes, diagrama de caso de uso, diagrama de componentes, diagrama de actividades y diagramas de secuencia; además de contar con una interfaz de usuario que brinda comodidad y eficiencia. Desde el punto de vista del fabricante, Poseidon For UML proporciona estabilidad, escalabilidad, rendimiento, fiabilidad, personalización, al igual que califican su interfaz de usuario como la mejor de la industria.

También cuenta con algunas potentes funciones como la ingeniería de ida y vuelta y la generación de documentación que se han implementado de forma inteligente sin la carga común a otras herramientas UML.

2.1.2 Principios Fundamentales. Los desarrolladores en cada actualización desde su versión inicial aportan a los usuarios calidad y eficiencia, especialmente en la versión Poseidon For UML 8.0 que se usa durante este proyecto, se rige bajo algunos principios fundamentales, los cuales son:

- **Pureza**

  Se centra en la productividad del modelador. La parte central de esta versión es el área de dibujo, ya que se enfoca en el trabajo del diagrama primordialmente, disminuyendo la interfaz de usuario, permitiendo al usuario un enfoque directo en el desarrollo.

- **Escalabilidad**

  Poseidon For UML ofrece diagramas pequeños de fácil uso que permite a usuarios que están iniciando el aprendizaje de la herramienta no conocer mucho sobre esta, buscando así que Poseidon For UML sea una herramienta mucho más rápida que un tablero o el típico papel y lápiz. De igual manera para modelos más grandes Poseidon For UML posee internamente una base de datos orientada a objetos.

- **Rendimiento**

  Ellos saben que una clave para la productividad es el rendimiento, por eso tiene gran cuidado en ofrecerlo siempre. En la nueva versión Poseidon For UML funcionan normalmente y disminuyen a la mitad del tiempo su marcha.

2.2.3. Versiones.

- **Community Edition:** Esta es una versión realizada para el aprendizaje y la enseñanza de Diagramas UML. Ofrece diferentes beneficios y ventajas para los usuarios de esta herramienta.

- **Starter Edition:** Versión Optimizada para profesionales en desarrollo y análisis, contiene todas las ventajas de la Community Edition mejoradas, incluyendo Ingeniería Inversa para Java, auto-Documentación, y extensiones.
- **Profesional Edition:** Diseñada para enfrentar las necesidades del desarrollador de Software Profesional, Incluye Ingeniería de ida y vuelta, Importación de JAR e integración con Eclipse IDE.

- **Emmbedded Edition:** Diseñada específicamente para el desarrollo de Sistemas embebido.

2.2.4 Licencia. Poseidon For UML cuenta con una licencia semipública, para decirlo en términos coloquiales, es decir su descarga e instalación es absolutamente libre, pero ya sus propiedades son con una suscripción de alquiler, los gastos fijos se incurren. El usuario puede elegir un período mensual, trimestral o anual para los pagos y uso.

Para este Estudio se usa a Poseidon For UML Community Edition, puesto que es una herramienta paga, y el presupuesto es bajo, se renta la licencia por medio de subscripción de esta edición por un mes, con un valor de 5 dólares.

2.2 STARUML

2.2.1 Introducción. Es un proyecto de software libre, que consiste en crear una herramienta de modelado de Software y plataforma que pretende ser una opción convincente versus las herramientas comerciales de UML. StarUML es un proyecto de código abierto, y según sus desarrolladores, rápido, flexible, con características extensibles, y de libre acceso-UML / MDA. Está ejecutado en plataforma win32.

StarUML está siendo constantemente actualizado con el fin de soportar cada una de las nuevas versiones de UML. También posee la nueva tecnología introducida por OMG, llamada MDA. StarUML está diseñada para servir de apoyo a la MDA y ofrece variables, como la personalización del perfil UML, enfoque, el marco del modelo, NX (extensión de la notación), código de MDA y de plantilla de documento, entre otros.

2.2.2 Principios Fundamentales. Ofrecer un amplio grupo de diagramas de UML 2.0, entre los cuales están: Diagrama de casos de uso, diagrama de clases, diagrama de secuencia, diagrama de colaboración, diagrama de estados, diagrama de actividad, diagrama de componentes, diagrama de despliegue, diagrama de estructura compuesta (UML 2.0). Al igual que soporta varios lenguajes entre los cuales se encuentra Java, C++, C# (generador de código y de ingeniería inversa). También genera documentos tipo Microsoft Office y códigos personalizable por el usuario y archivos de comandos activados (JScript), y maneja una alta compatibilidad.

StarUML ofrece en su edición diálogos rápidos, atajos de comandos, múltiples deshacer / rehacer y manipulación del teclado, y en la interfaz de usuario VS.NET apariencia y ventanas acoplables.

Según los desarrolladores y la teoría lo afirma la usabilidad es lo mas importante en el desarrollo de software, por tal motivo StarUML es implementado con el fin de proporcionar características “Amigables” al usuario como los diálogos rápidos, la manipulación de teclado, la descripción de diagramas, entre otros.

2.2.3 Versiones. La primera versión (v0.9) de StarUML o plástico -anteriormente era conocida como "Plastic" o "de Plastic Agora"- era una herramienta sencilla que se utilizaba para extraer los módulos de software y sus dependencias. En 1997 fue liberado y llamado Platico 1.0 Liberado, en 2003 fue De platico 2003 también liberado, que era una herramienta completamente rediseñada y reescrita, compatible con UML 1.4 y de arquitectura abierta. Solo en el 2005 cambia su nombre a lo que se conoce actualmente siendo llamada StarUML 5.0 y liberado totalmente.

2.2.4 Licencia. Los Módulos StarUML y el núcleo se encuentran bajo los términos de la GPL (GNU Public License) con las siguientes dos excepciones[4]:

1. Permiten vincular varias bibliotecas comercial específico y *los componentes.* (Esta excepción es una decisión inevitable que a su vez StarUML, anteriormente un producto comercial, como un software de código abierto. Sin embargo, estas bibliotecas y componentes son relativamente populares y no son costosos. En el largo plazo, están dispuestos a reemplazarlo con las cosas de código abierto.)[5]

2. Deja que se enlacen los plug-in de propiedad módulos. *(Esta excepción es para las personas* que quieren vender comercialmente los módulos plug-in que se ejecutan en la plataforma StarUML. Esto permitirá la ampliación de los

[4] Página oficial del distribuidor http://staruml.sourceforge.net/en/license.php

desarrolladores y usuarios de la comunidad y la producción de tecnologías de mayor valor y productos relacionados con UML y MDA)[6].

Para este Estudio se usa Star UML versión 5.0, ya que es la última versión actualmente operativa y estable de la herramienta de libre distribución.

2.3 ENTERPRISE ARCHITECT

2.3.1 Introducción. Es una plataforma de modelado UML integral de herramientas de análisis y diseño, con código de ingeniería para más de 10 idiomas. Cuenta con un buen soporte para modelar negocios, software y sistemas. Según sus desarrolladores, esta herramienta cuenta con una trazabilidad completa desde los requisitos hasta la implementación y escalabilidad, características serán probadas durante el desarrollo de este proyecto. También proporcionan mapas mentales, estructurales empresariales, notación para el Modelado de Procesos de Negocio (BPMN), entre muchos más.

2.3.2 Principios Fundamentales. Esta herramienta está construida sobre las especificaciones de UML.2, pero a demás de eso utiliza los perfiles UML para extender el dominio de modelado, mientras asegura la integridad.

Enterprise Architect provee modelado de ciclo de vida completo para:

- Negocios y los sistemas de TI.
- Software e Ingeniería de Sistemas.
- El desarrollo en tiempo real y embebido.

Con capacidades integradas de gestión de requisitos, Enterprise Architect ayuda a trazar especificaciones de alto nivel a los modelos de análisis, diseño,

[6] http://staruml.sourceforge.net/en/license.php

implementación, prueba y mantenimiento, utilizando UML, SysML, BPMN y otros estándares abiertos. Es una herramienta multiusuario, una herramienta gráfica diseñada para ayudar a sus equipos a desarrollar sistemas robustos y de fácil mantenimiento.

Enterprise Architect soporta la generación e ingeniería inversa del código fuente para muchos lenguajes populares, incluyendo: Action Script, Ada, C y C + +, C #, Java, Delphi, Verilog, PHP, VHDL, Pitón, VB.Net, Visual Basic, y más...

Entre sus principios este el de contar con un editor de código incorporado le permite navegar rápidamente, en el mismo entorno directamente en el código fuente, también cuenta con plantillas de generación de código que le permiten personalizar lo que se genero, de acuerdo a las especificaciones de su compañía.

Enterprise Architect ofrece a los gerentes de proyectos la opción de asignar recursos, implementar procedimientos de control de cambio y mantenimiento, medir riesgos y esfuerzos, estimar tamaño y complejidad.

Velocidad, estabilidad y rendimiento, puesto que Enterprise Architect considerado un artista interpretando o ejecutante con rapidez al cargar los modelos extremadamente grandes en cuestión de segundos. Con un repositorio de modelos de alto rendimiento, Enterprise Architect se adapta fácilmente a grandes equipos compartiendo la misma visión de la empresa. Con capacidad de control de versiones estrechamente integradas, al igual que permite tener equipos distribuidos a nivel global para colaborar eficazmente en proyectos comunes[7].

Enterprise Architect ofrece a los gerentes de proyectos la opción de asignar recursos, implementar procedimientos de control de cambio y mantenimiento, medir riesgos y esfuerzos, estimar tamaño y complejidad.

[7] REFERENCIA DE LA PAGINA DE LOS DESARROLLADORES http://www.sparxsystems.com/products/ea/index.html

2.3.3 Versiones. La integración de muchas características de gama alta para los ingenieros de sistemas, las ediciones ultímate y de ingeniería de sistemas de Enterprise Architect proporciona compatibilidad integrada por: Sysml 1,1,1,2, parámetro de simulación del modelo, la generación de código ejecutable, modelo a las transformaciones de código para los lenguajes de descripción de hardware y el ada 2005.

2.3.4 Licencia. Enterprise Architect es vendido como un producto licenciado de acuerdo con los términos y condiciones de contrato de licencia de usuario final (CLUF). Comprar una licencia lo convierte en un usuario registrado por 12 meses, lo cual le da derecho a:

- Descargar y activar la actual versión completa de EA.
- Acceder a actualizaciones y nuevas compilaciones gratuitamente por un período de 12 meses.
- Acceder al soporte en Español de Sparx Systems por 12 meses.
- Acceder a la sección de usuarios registrados y a cualquier recurso asociado por 12 meses.

En la actualidad el programa de ingeniería de sistemas de la Universidad Tecnológica de Pereira adquirió licencias de la herramienta Enterprise Architect, por tal motivo decidimos hacer uso de estas, y se realizan los diagramas en Enterprise Architect versión 7.0.

3. CASO DE ESTUDIO

Se presenta el siguiente caso hipotético:

Los Estudiantes del programa Ingeniería de Sistemas y Computación deben cumplir con los siguientes requisitos para aspirar a su título profesional:

- Aprobar 188 créditos correspondientes a cursar y aprobar las asignaturas que exige la facultad.
- Realizar el Taller de Símbolos
- Aprobar los niveles necesarios de Ingles y/o aprobar el examen de suficiencia de Ingles.
- Presentar las pruebas SABER PRO.
- Presentar y Obtener una aprobación en el proyecto de grado.

Se pretende realizar una herramienta web, por medio de la cual los estudiantes puedan verificar el estado de dichos requisitos de graduación. Además el estudiante también podrá consultar los pasos a seguir para cumplir con estos a fin de generar mayor comodidad para el estudiante y no encontrarse con contratiempos a la hora de aspirar al título profesional.

Para desarrollar este caso de estudio se requerirá realizar los siguientes pasos:

1. Determinar y definir las preguntas importantes para la investigación.
2. Seleccionar el caso de estudio y determinar las técnicas de evaluación.

3. Recopilar información de las herramientas usadas.
4. Evaluar y analizar los Datos.
5. Realizar un reporte con los resultados obtenidos.

El objetivo en este estudio es evaluar las ventajas y desventajas, por medio de métricas, que existen entre las herramientas CASE.

Dicho esto se realizará el modelamiento UML en la etapa de análisis y diseño del software para un sistema que permita realizar el caso de estudio propuesto en las 3 herramientas CASE que se están estudiando (STARUML, POSEIDON FOR UML Y ENTERPRISE ARCHITECT). De esta manera se podrá realizar el estudio comparativo con el fin de obtener una evaluación de las mismas.

Por medio de los siguientes pasos se identifican las necesidades que busca satisfacer el software a modelar.

1. Determinar y definir las preguntas importantes para la investigación.

- ¿Qué actores incluirá el modelo de este sistema?
- ¿Cuáles son las clases que se implementarán?
- ¿Cuál es la importancia de cada uno de los módulos?
- ¿Cómo afecta a la integridad del sistema un cambio en el mismo?

2. Para verificar el caso de estudio como tal, se deben tener en cuenta las diferentes tareas que debe realizar el estudiante en el momento de solicitar su proyecto de grado asumiendo que de manera correcta ha cumplido con cada uno de los requisitos necesarios para aspirar a su título profesional.

   Se determinarán las clases, los actores del sistema y las métricas por medio de las cuales se realizará la evaluación comparativa entre ellas mismas y

finalmente se realizará la tabla de evaluación donde se observarán los resultados de toda la investigación, que plasma cada una de las características entre ellas medidas desde el punto de vista de las métricas que serán definidas en el trascurso de este documento.

3.1 DEFINICIÓN DE LOS ACTORES

- El actor principal en este sistema será el estudiante, puesto que es el responsable del cumplimiento de los requisitos para aspirar a su título profesional. Además es quien debe tomar decisiones y acciones que le permitan cumplir con estos objetivos.

- El segundo actor en ese caso de estudio sería la universidad como tal, es decir, el funcionario encargado de verificar, con la ayuda del sistema, que el estudiante cumpla con todos los requisitos y cambie el estado del estudiante como apto para la obtención del título, de manera que este pueda hacer su solicitud de grado.

3.1.1 Definición De Las Clases. Entre las clases se pueden ubicar los requisitos que debe cumplir el estudiante como acciones que debe realizar. Entre ellas se tiene:

- Aprobar Créditos.
- Taller de Símbolos.
- Realizar Pruebas SABER PRO.
- Demostrar conocimiento del inglés.
- Aprobación del proyecto de grado.

4. MODELAMIENTO DEL CASO DE ESTUDIO

En este punto se puede iniciar con el desarrollo del modelo respectivo para cada uno de los diagramas que se realizarán:

- Modelo de casos de uso.
- Diagrama de actividades.
- Diagrama de clases.
- Diagrama de secuencia.
- Diagrama de colaboración.
- Diagrama de estados.
- Diagrama de componentes.

Los pasos **3**, **4** y **5**, se desarrollan en los próximos capítulos, ya que se hace necesario el análisis de las herramientas.

4.1 DESCRIPCIÓN DEL MODELO DE CASO DE USO

En esta parte del documento se genera la siguiente descripción de manera detallada de las transacciones que conforman los casos de uso. La descripción indica los pasos concretos que realiza el sistema en el momento en que el actor realiza determinada actividad interactuando con el sistema y cuando este le entrega información al actor. Aquí se describen las verificaciones omitiendo los aspectos de manejo de errores.

Tabla 1. CASO DE USO APROBAR ASIGNATURA

Nombre del caso de uso		Aprobar de Asignaturas
Actores involucrados		Estudiante
Condiciones de Entrada		Matrícula financiera, Matrícula Académica
Condiciones de Salida		Créditos necesarios aprobados
Inclusiones		Ninguna
Serie de Pasos		
1	Comprobar que el estudiante se encuentra al día con la matricula financiera.	Sistema
2	Al final de cada semestre se comprueba cuantos créditos aprobó el estudiante.	Sistema
3	Si el número de créditos aprobados es igual al número de créditos propuestos por el programa se da una alerta avisándole al estudiante que ya ha aprobado las materias que propone el programa	Sistema
4	Revisar si se finalizaron o no las materias propuestas por el programa	Estudiante

Fuente: Autores

Tabla 2. CASO DE USO EXAMEN DE INGLES

Nombre del caso de uso		Examen de Ingles
Actores involucrados		Estudiante
Condiciones de Entrada		Matrícula financiera, Matrícula Académica
Condiciones de Salida		Aprobación del examen con el 75%
Inclusiones		Ninguna
Serie de Pasos		
1	El sistema importa la nota del estudiante.	Sistema
2	Se comprueba si el examen se aprobó con el 75% o más	Sistema
3	Si se aprobó, el sistema informará al estudiante que ha aprobado el examen. De lo contrario el sistema esperará a que el estudiante presente otra prueba	Sistema

Fuente: Autores

Tabla 3. CASO DE USO PRUEBAS SABER PRO

Nombre del Caso de Uso		Pruebas Saber Pro
Actores Involucrados		Estudiante, Administrador.
Condiciones de Entrada		El Estudiante debe estar cursando noveno semestre
Condiciones de Salida		El estudiante ya ha presentado las pruebas
Inclusiones		Ninguna
Serie de Pasos		
1	Verificar el Semestre actual del estudiante y en caso de que sea el noveno, dar aviso, para presentar Pruebas Saber Pro.	Sistema
2	Verificar si el estudiante ha presentado dichas pruebas por medio del documento que lo comprueba.	Administrador
3	Revisar si es momento de realizar pruebas saber pro.	Estudiante

Fuente: Autores

Tabla 4. CASO DE USO PROYECTO DE GRADO

Nombre del Caso de Uso		Proyecto de Grado
Actores Involucrados		Estudiante, Administrador
Condiciones de Entrada		Presentar Anteproyecto, Anteproyecto Aprobado, Estar Cursando o haber cursado Proyecto de Grado I
Condiciones de Salida		Nota Aprobada del Proyecto de Grado
Inclusiones		Ninguna
Serie de Pasos		
1	Verificar que el anteproyecto este aprobado	Administrador
2	Verificar que el Proyecto tenga un docente director	Administrador
3	Verificar Nota del Proyecto	Sistema
4	Consultar nota del proyecto	Estudiante

Fuente: Autores

Tabla 5. CASO DE USO OBTENER TITULO

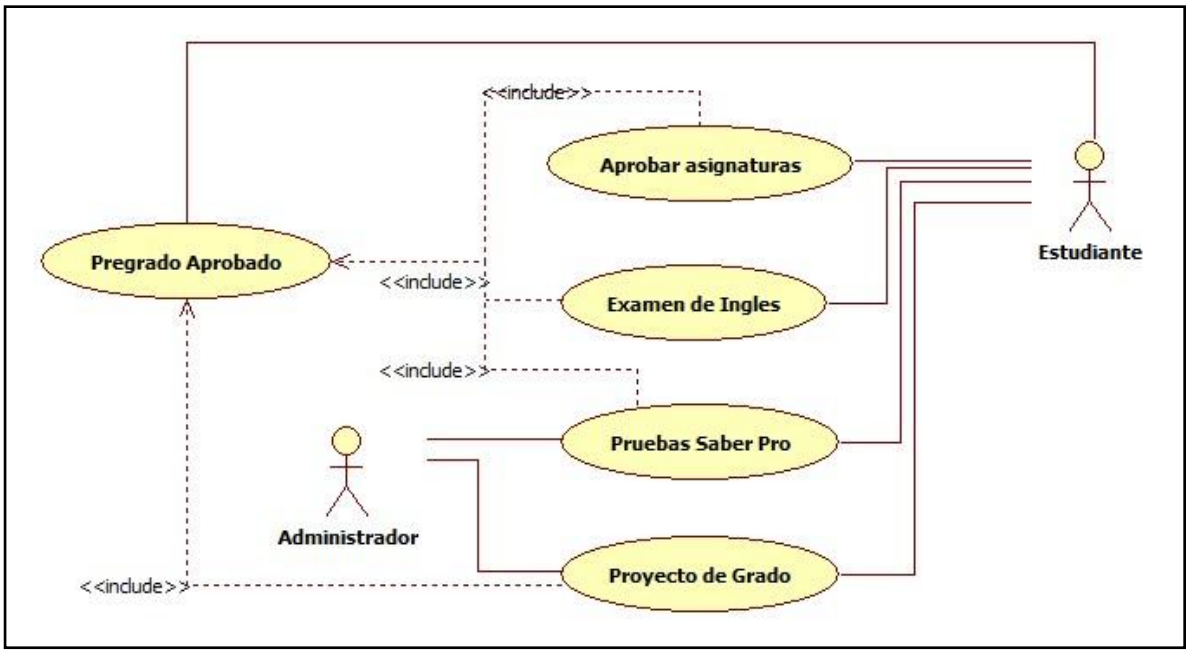
Nombre del Caso de Uso	Candidato al Titulo
Actores Involucrados	Estudiante
Condiciones de Entrada	Asignaturas aprobadas, Examen de Ingles, Pruebas Saber Pro Presentadas, Proyecto de Grado Aprobado.
Condiciones de Salida	Estudiante Graduado
Inclusiones	Proyecto de Grado, Pruebas Saber Pro, Aprobar de Asignaturas, Examen de Inglés (estas son las rutinas descritas anteriormente)

Fuente: Autores

4.2 DIAGRAMA DE CASOS DE USO

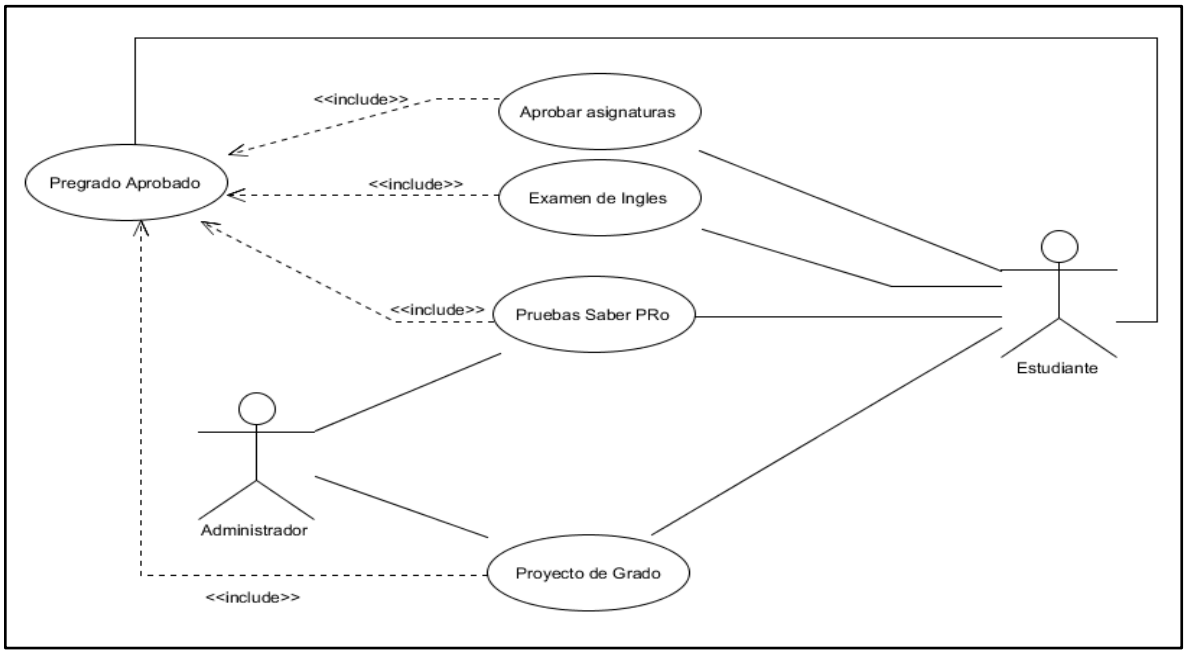
En el modelo de casos de uso se establecieron las principales interacciones por parte del sistema con agentes externos al mismo, dejando como conclusión aquellas funciones principales por parte de este. Es así como, al darle una interpretación grafica al modelo por medio de diagramas UML, se obtienen el diagrama de casos de uso que se muestra (**figura 1.1, 1.2 y 1.3**), los cuales corresponden a el mismo diagrama de casos de uso pero implementado a través de las tres herramientas StarUML, Poseidon For UML y Enterprise Architect respectivamente.

Figura 1.1: Diagrama de Casos De Uso StarUML



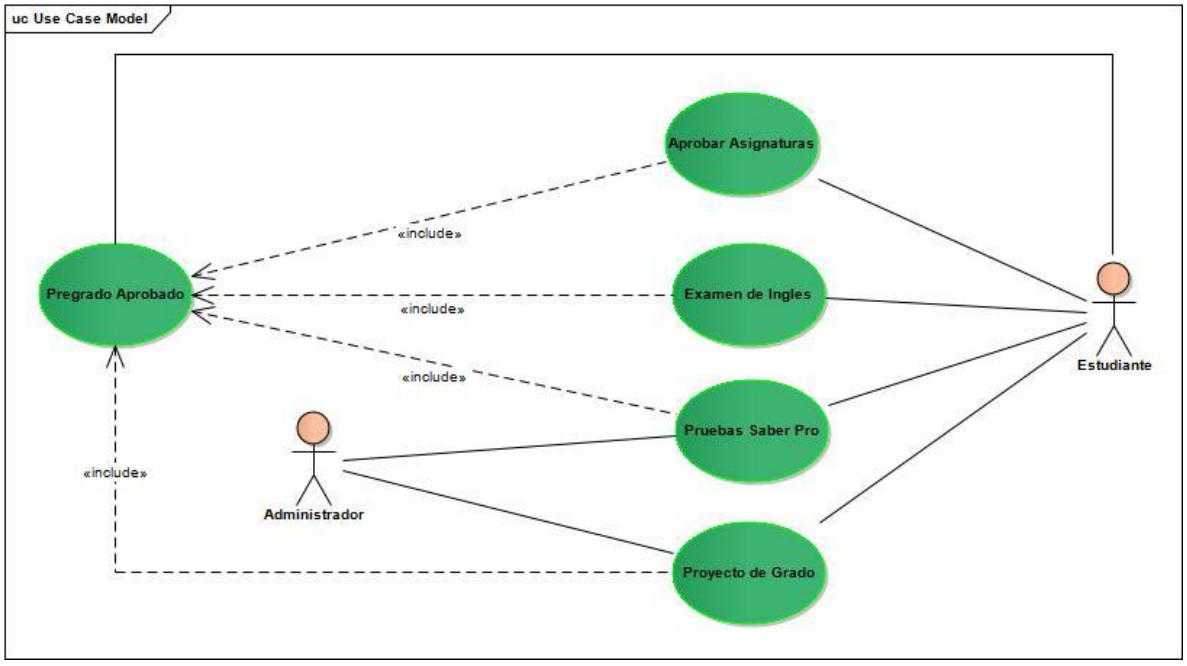
Fuente: Autores

Figura 1.2: Diagrama de Casos de Uso Poseidon For UML



Fuente: Autores

Figura 1.3: Diagrama de Casos de Uso Enterprise Architect



Fuente: Autores

4.3 DIAGRAMA DE CLASES

En desarrollo de software estos diagramas se usan para tener una visión de cómo se relacionan las clases que hacen parte del sistema, sus atributos y métodos.
Se considera prudente definir algunos conceptos con el fin de no dejar interrogantes en el aire, estos son:

- ***Clases***

  Las clases son la unidad básica. Contienen las características de los objetos y por medio de ellos se puede describir gráficamente el sistema.

- **Objetos**

  Los objetos son instancias de las clases. Cada objeto es creado a partir de las descripciones dadas en las clases a las que pertenecen.

- **Atributos**

  Los atributos son aquellas características que tienen los objetos, definidos en sus clases.

- ***Métodos***

  Son aquellas operaciones que pueden realizar los objetos de la clase.

De esta forma, y dejando en claro estos conceptos, se pasa a determinar qué elementos del modelo de casos de uso se pueden convertir en una clase con métodos que trabajen en el objetivo que busca el sistema que se está desarrollando.

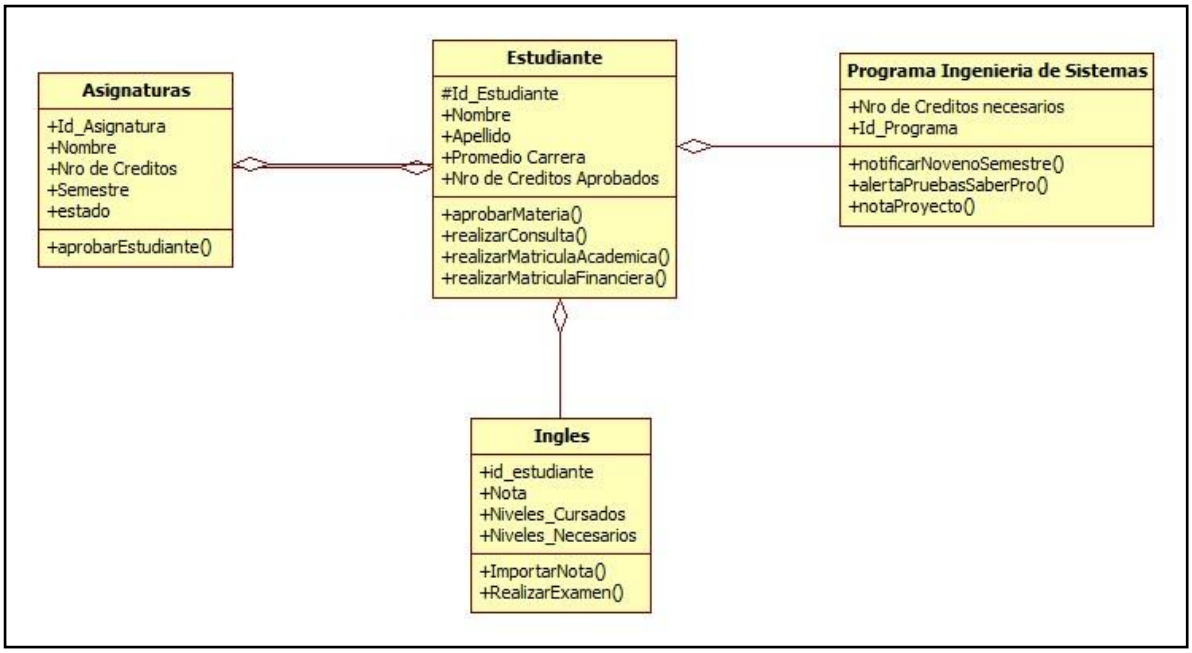
4.4 DETERMINACIÓN DE CLASE

Para determinar cuáles son las clases que se deben usar en el diagrama de clases, hay que identificar los sustantivos que se utilizaron en la descripción de los casos de uso, ya que estos son los que pueden convertirse en clases del sistema. Se realizó la siguiente lista:

- **Clase:** Estudiante **Decisión:** Si
- **Clase:** Programa Ingeniería de sistemas **Decisión:** Si
- **Clase:** Asignatura **Decisión:** Si
- **Clase:** Examen **Decisión:** No

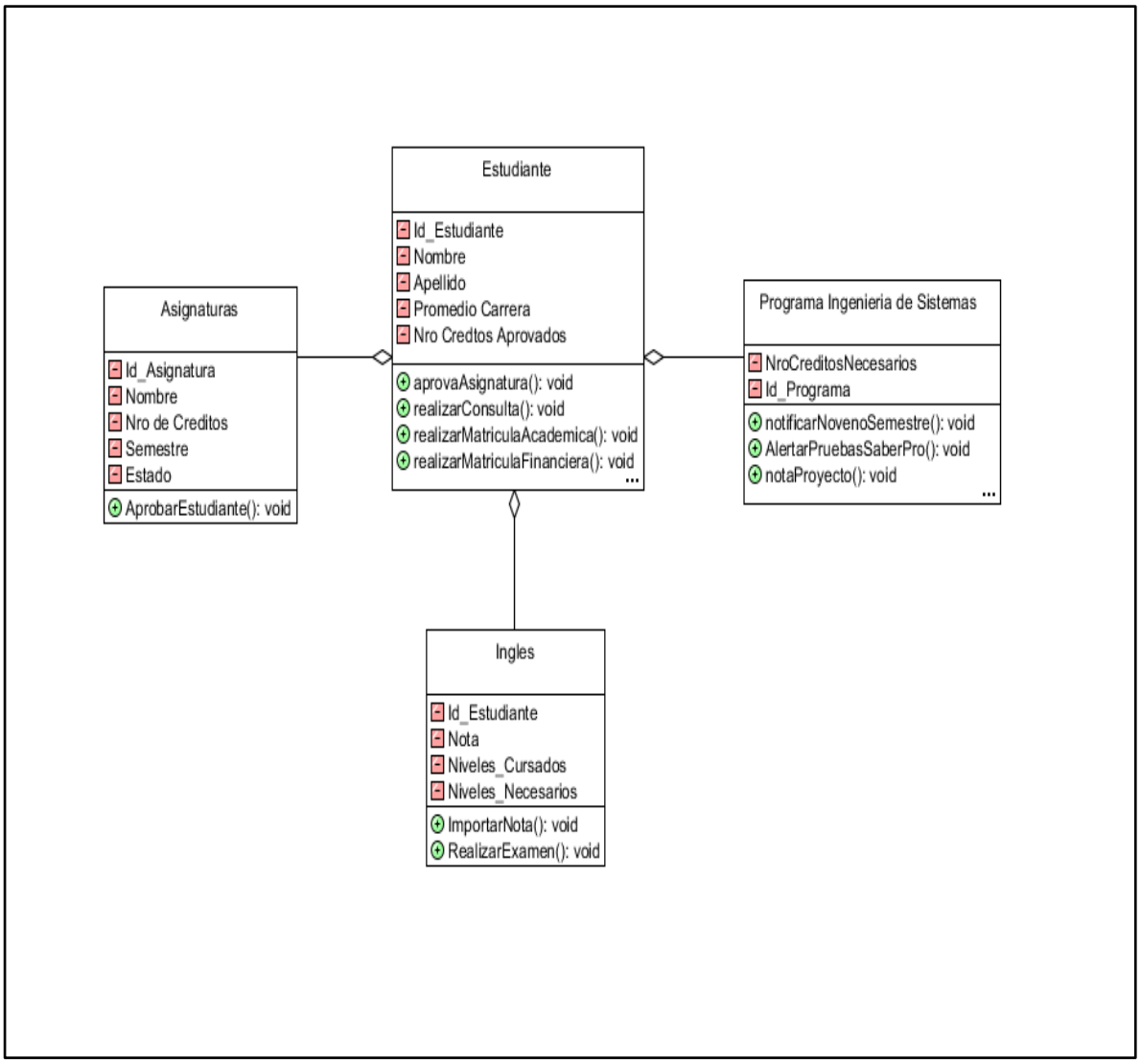
Se realiza entonces el siguiente modelo de clases en el cual se describen las clases que hacen parte del sistema y las relaciones entre ellas (***Figura 2.1, 2.2, 2.3***) el cual corresponde a los diagramas en las tres herramientas.

Figura 2.1: Diagrama de Clases StarUML



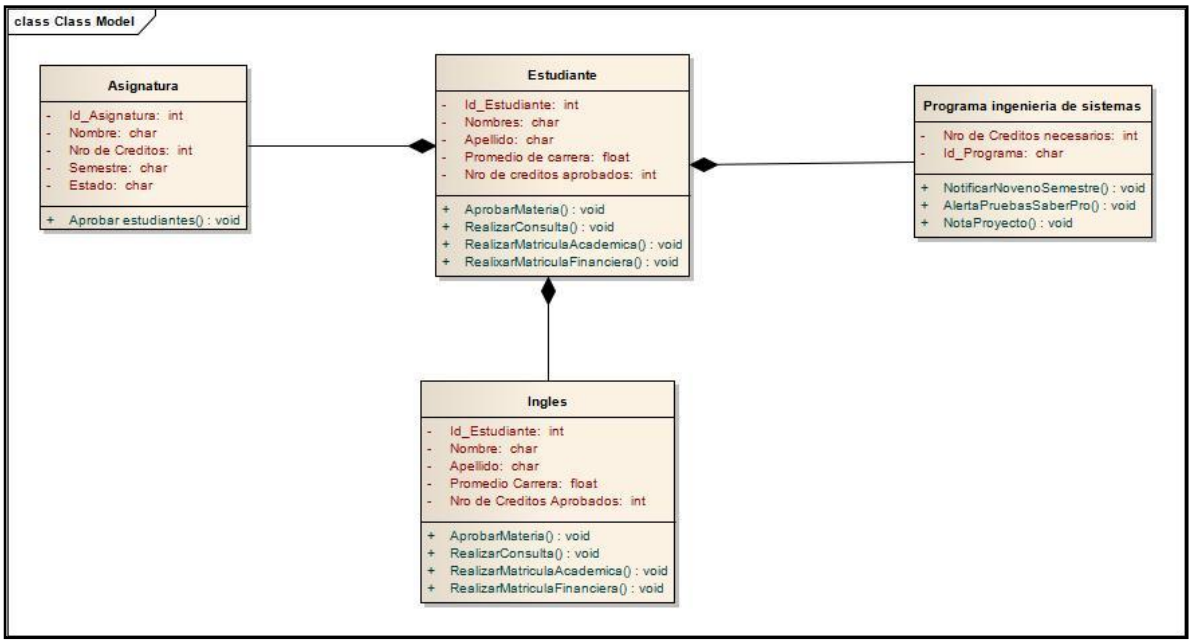
Fuente: Autores

Figura 2.2: Diagrama de Clases Poseidon For UML



Fuente: Autores

Figura 2.3: Diagrama de Clases Enterprise Architect



Fuente: Autores

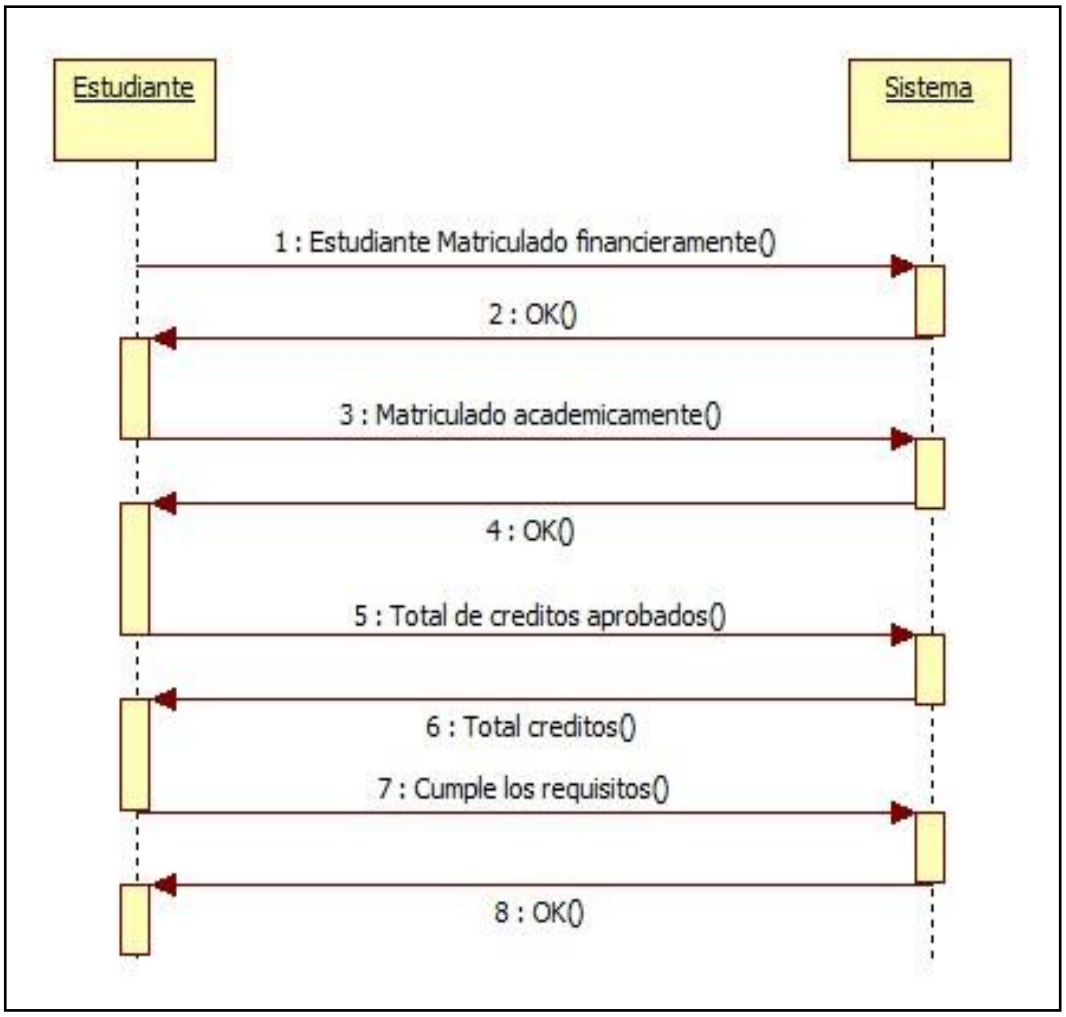
4.5 DIAGRAMAS DE SECUENCIA

Los diagramas de secuencia modelan la interacción a través del tiempo entre los objetos. Se pueden realizar diagramas de secuencia por cada caso de uso o sub-casos, que son una parte del caso de uso.

Secuencias para el caso de uso Asignatura

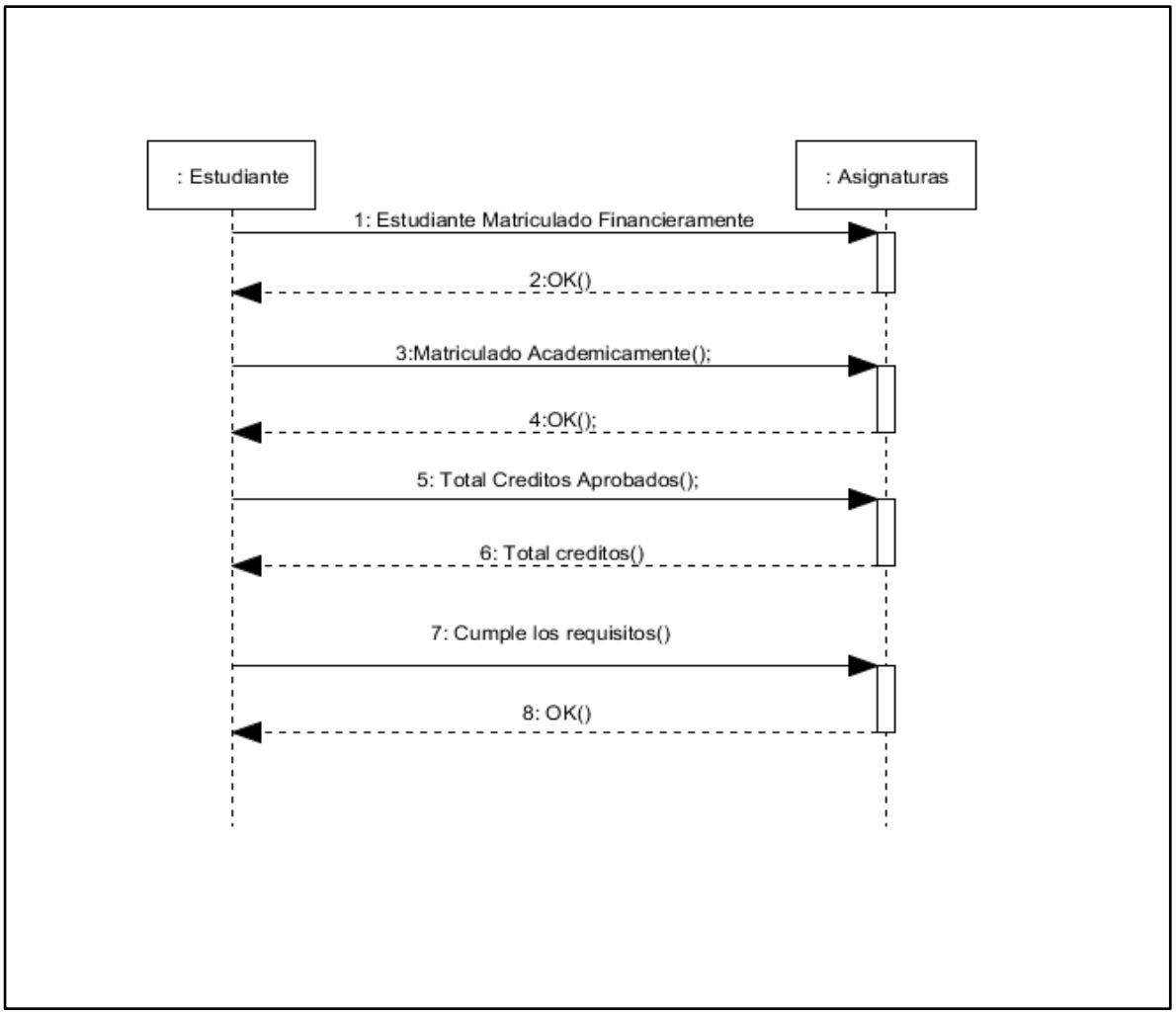
En este se puede ver la interacción entre el estudiante y el sistema en el caso de uso Asignatura **(Figura 3.1, 3.2, 3.3)**.

Figura 3.1: Diagrama de Secuencias Asignaturas StarUML



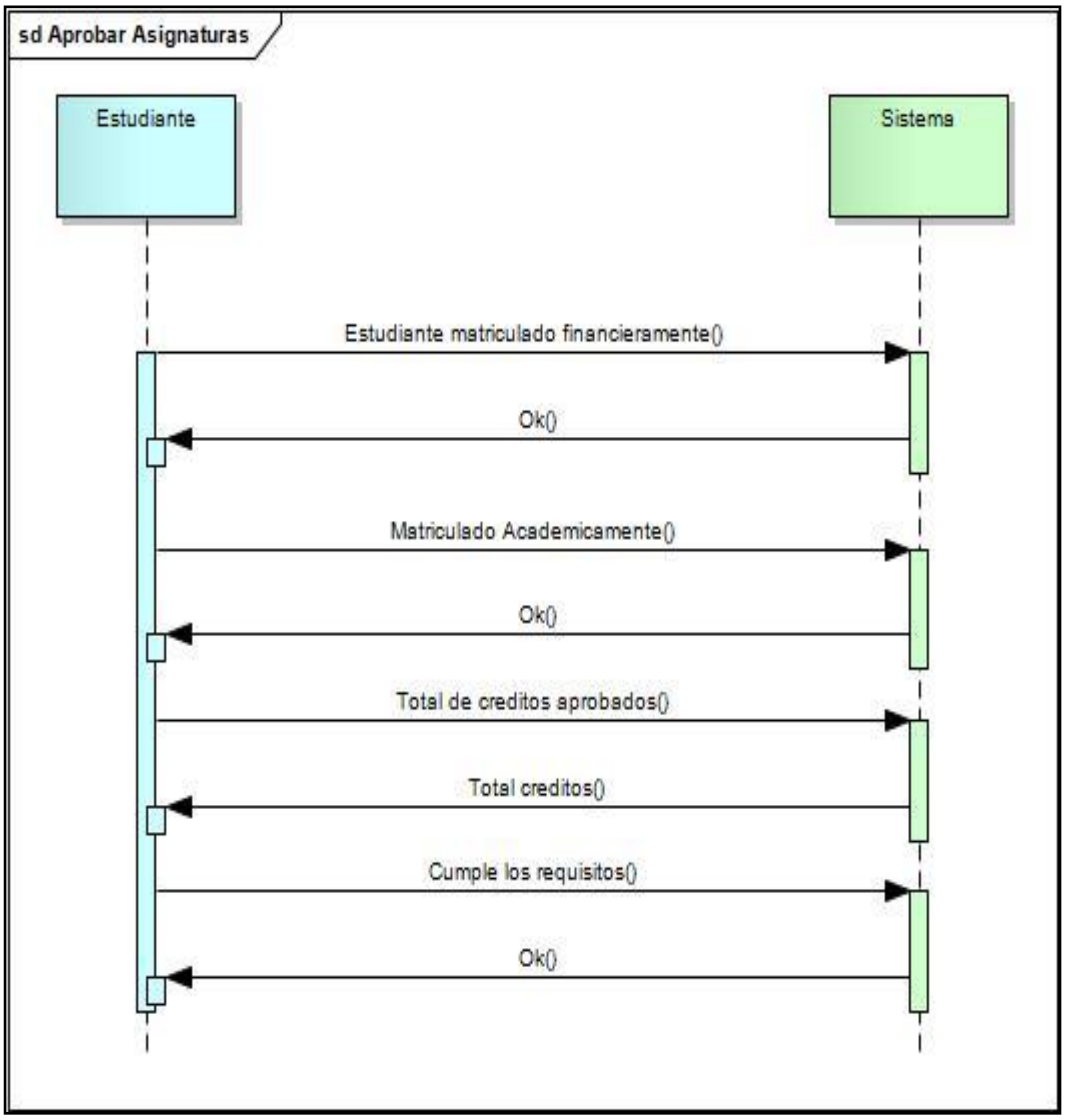
Fuente: Autores

Figura 3.2: Diagrama de Secuencias Asignaturas Poseidon For UML



Fuente: Autores

Figura 3.3: Diagrama de Secuencias Asignaturas Enterprise Architect

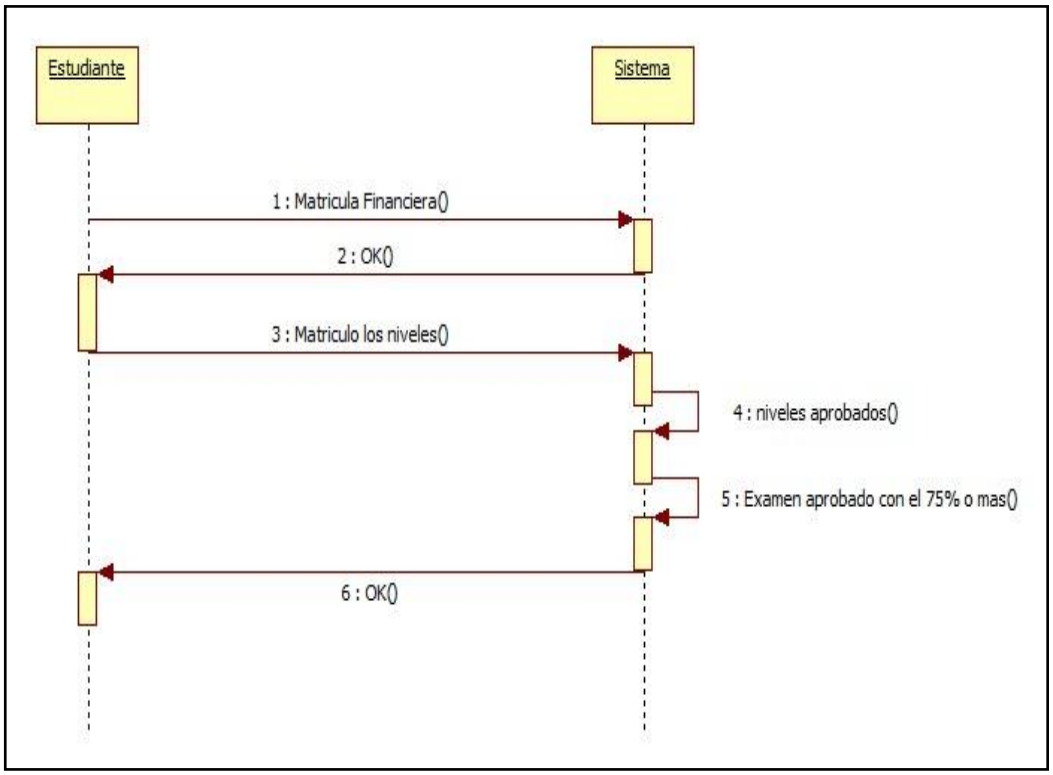


Fuente: Autores

Secuencias para el caso de uso Ingles

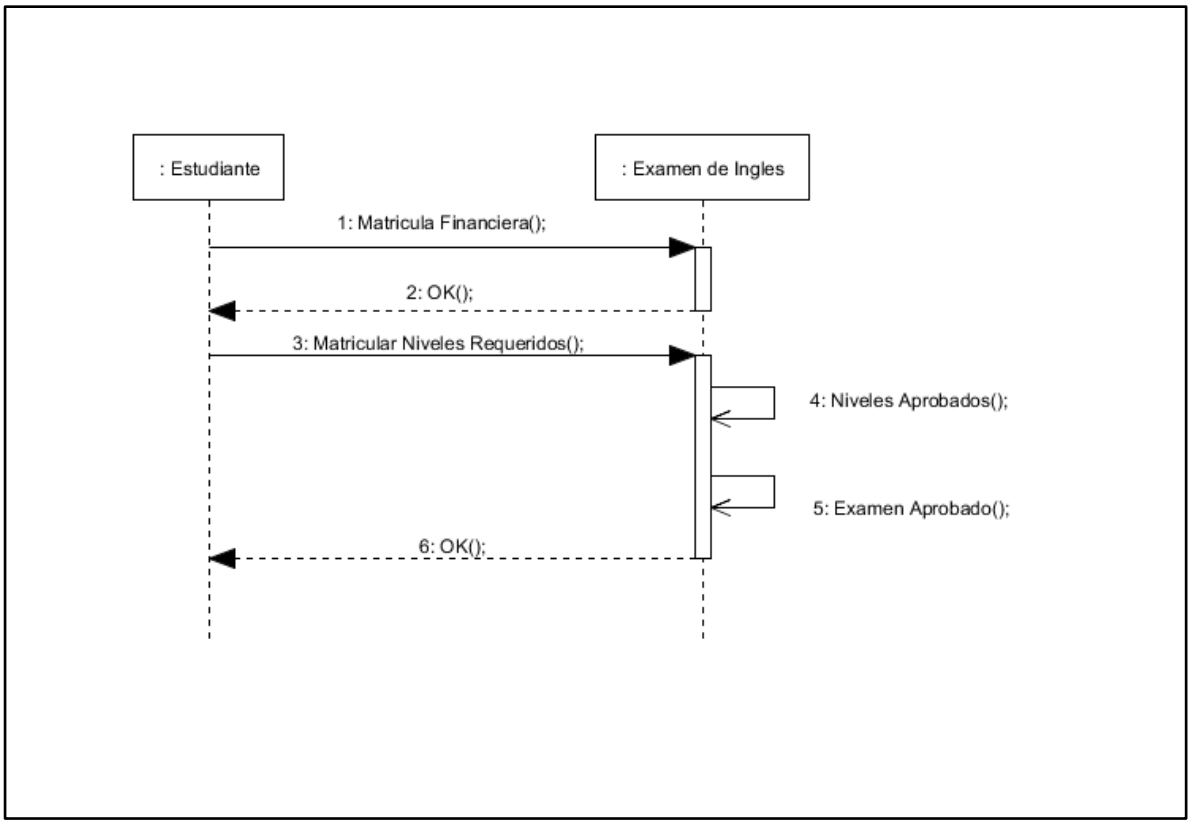
Se puede observar el siguiente diagrama que al igual que los otros es implementado e todas las herramientas evaluadas **(Figura 4.1, 4.2, 4.3)**, y permite observar cómo están interactuando el estudiante y el sistema durante el caso de uso Examen Inglés.

Figura 4.1: Diagrama de Secuencias Ingles StarUML



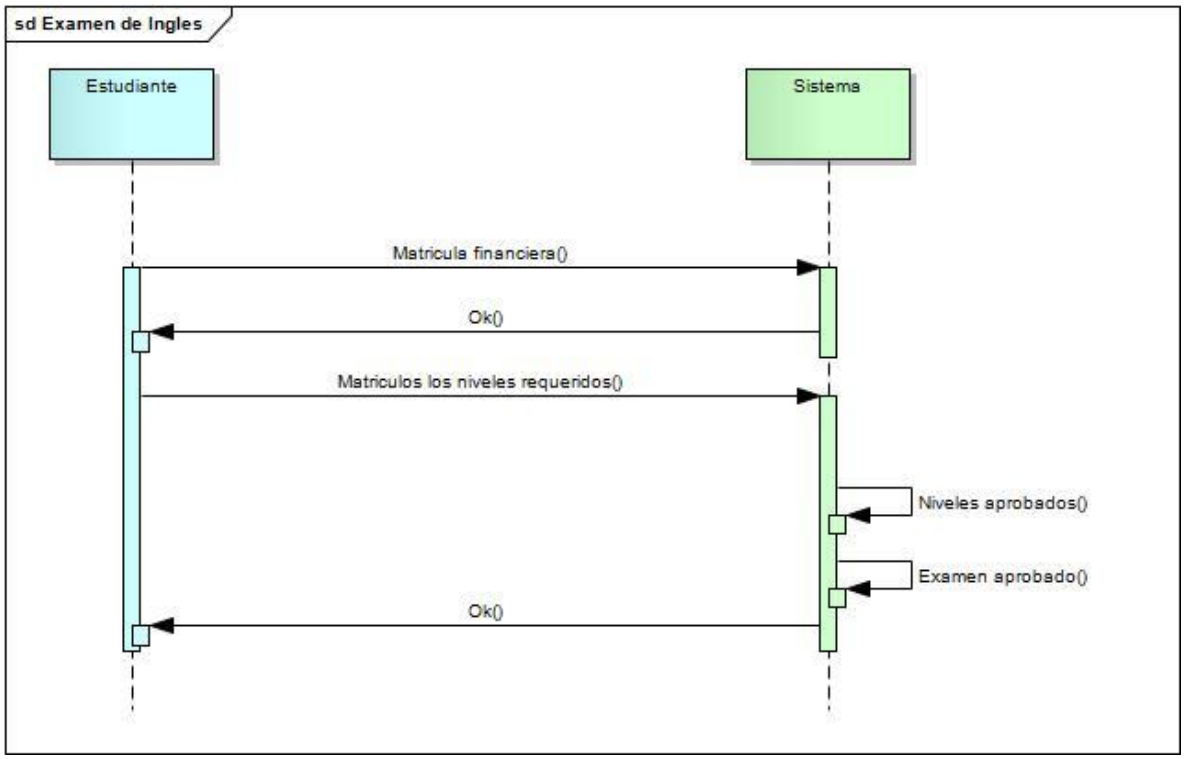
Fuente: Autores

Figura 4.2: Diagrama de Secuencias Ingles Poseidon For UML



Fuente: Autores

Figura 4.3: Diagrama de Secuencias Ingles Enterprise Architect

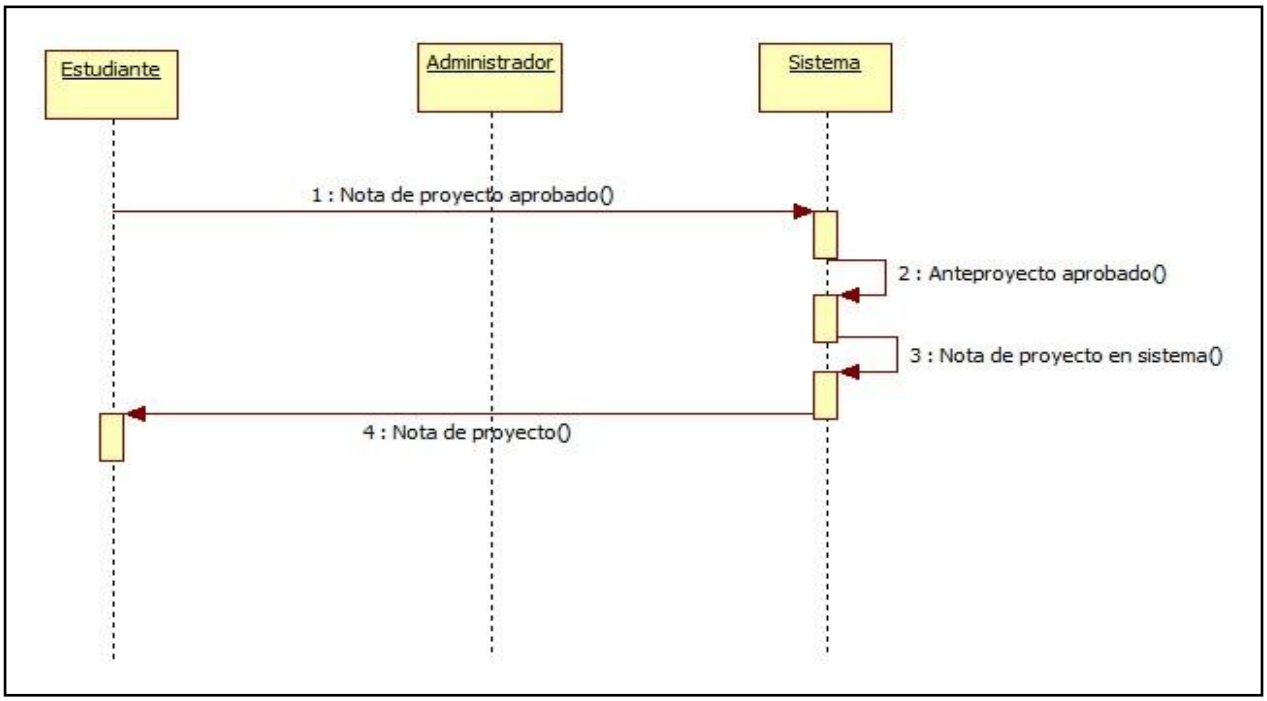


Fuente: Autores

Secuencias para el caso de uso Proyecto de Grado

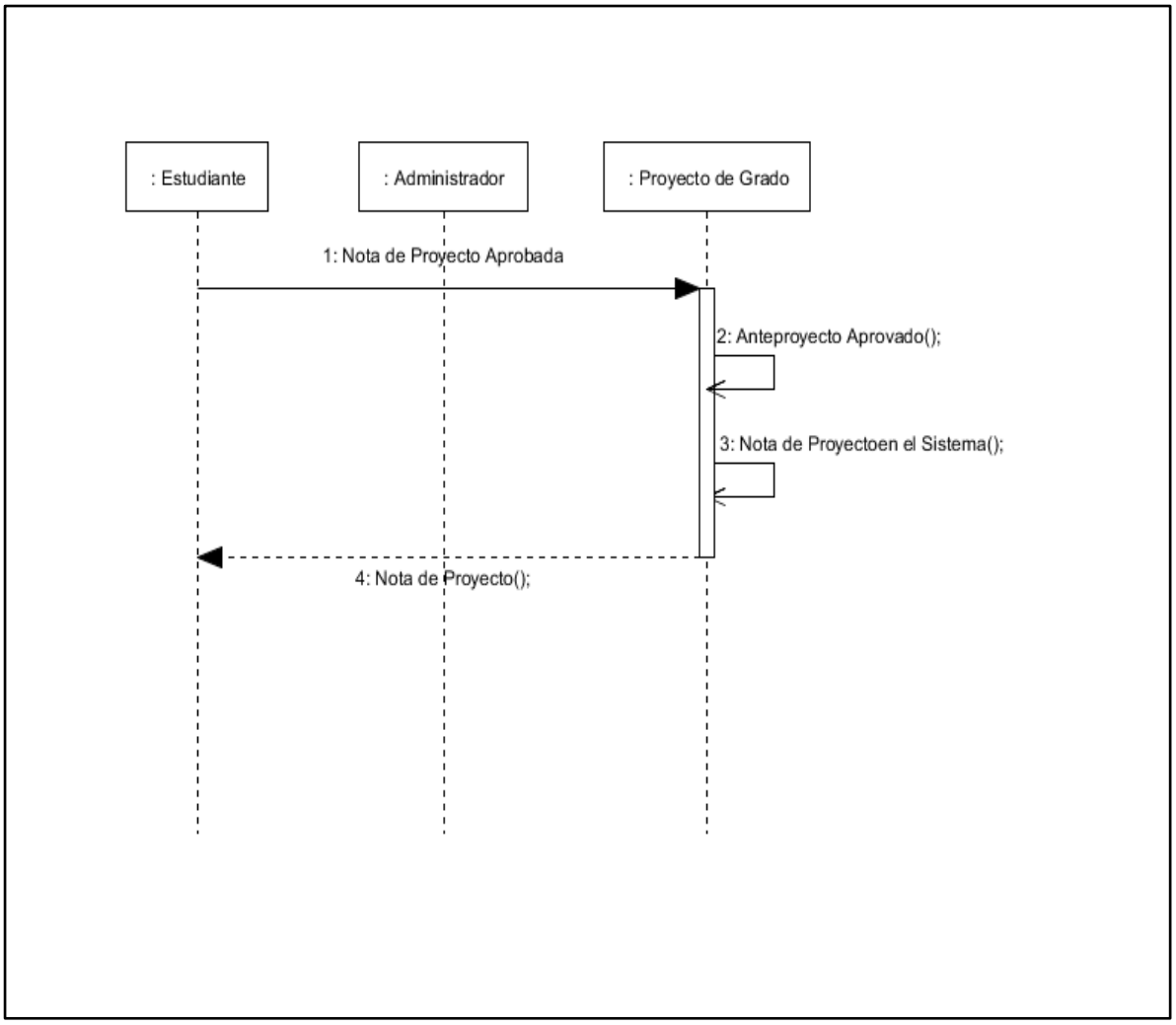
Se puede observar cómo interactúan los dos actores estudiante-administrador con el sistema durante el caso de uso Proyecto de grado (***Figura 5.1, 5.2, 5.3***).

Figura 5.1: Diagrama de Secuencias Proyecto de Grado StarUML



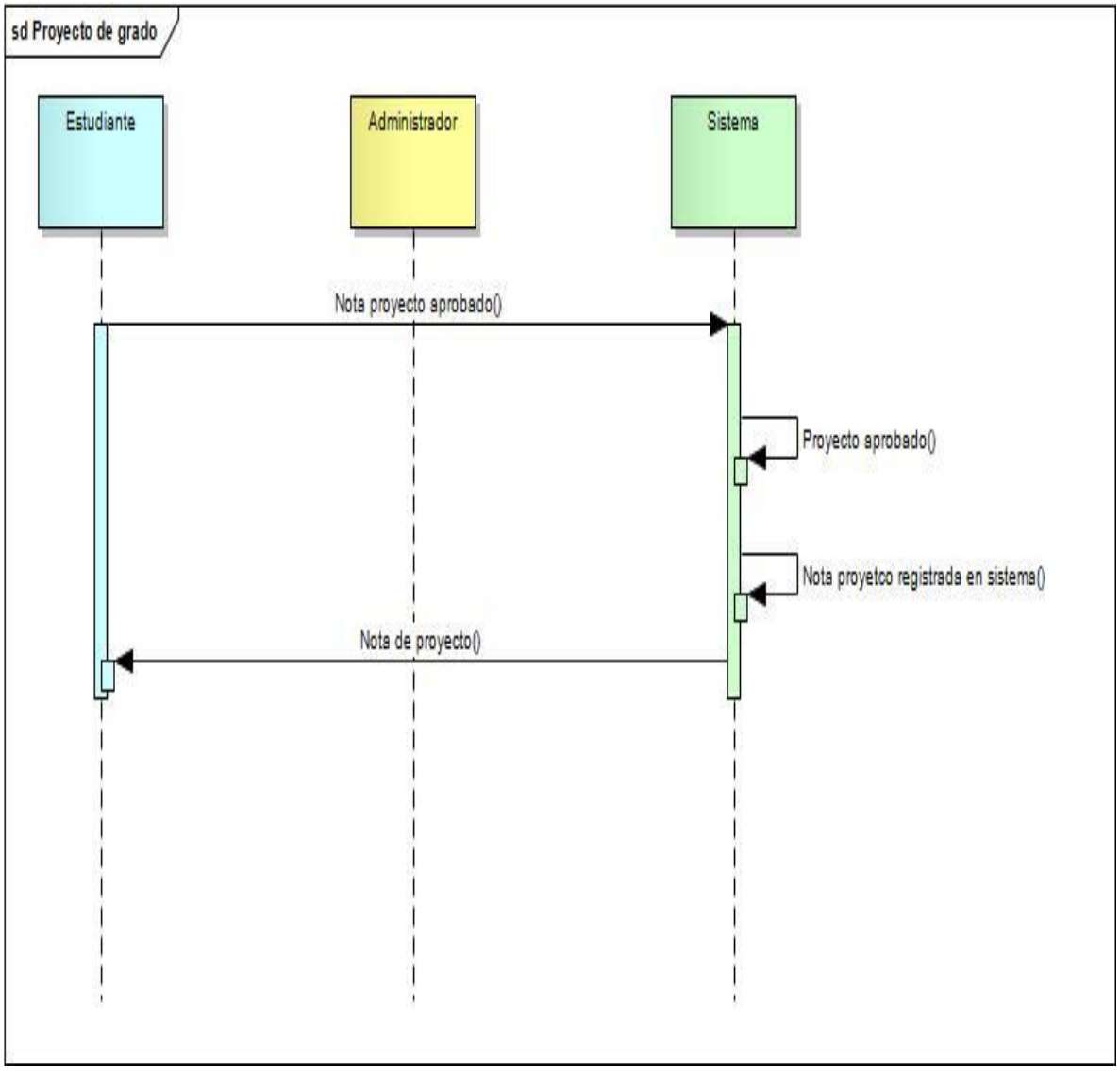
Fuente: Autores

Figura 5.2: Diagrama de Secuencias Proyecto de Grado Poseidon For UML



Fuente: Autores

Figura 5.3: Diagrama de Secuencias Proyecto de Grado Enterprise Architect

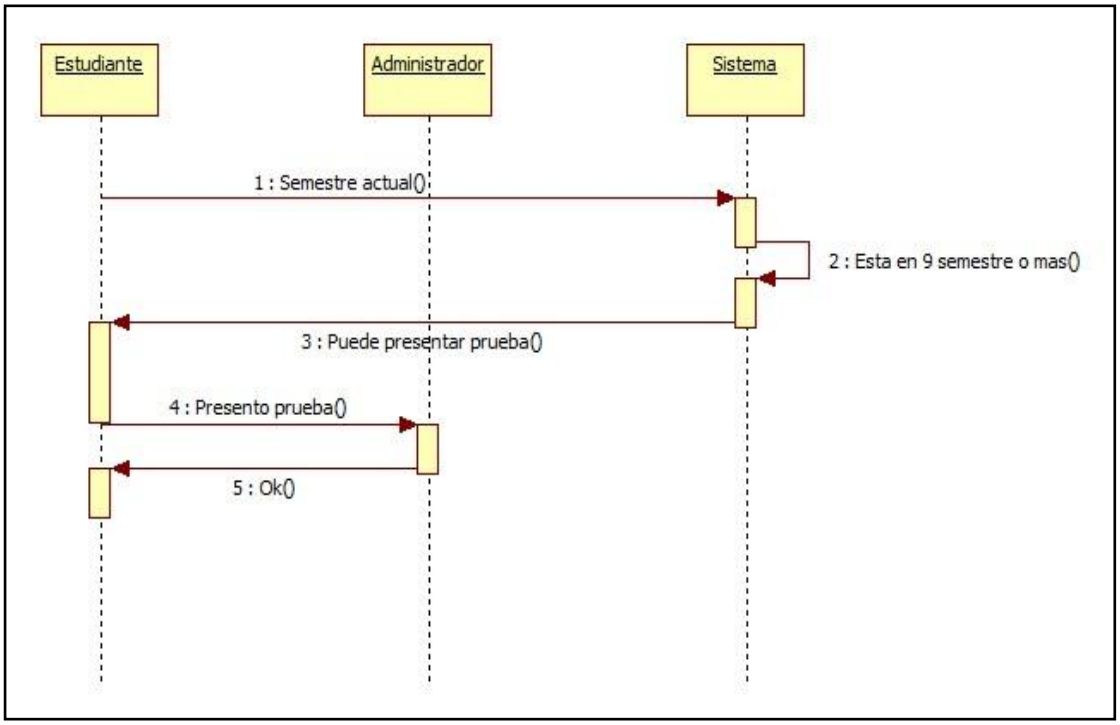


Fuente: Autores

Secuencias para el caso de uso Pruebas Saber Pro

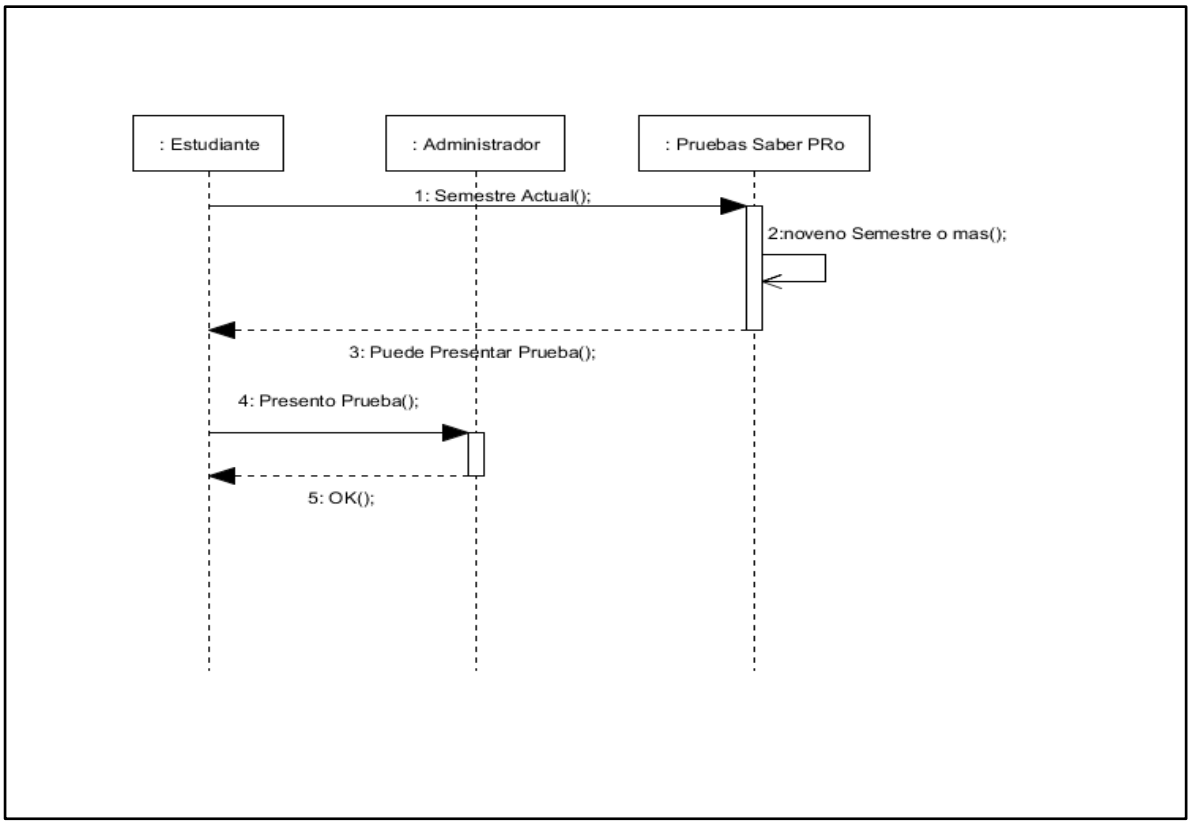
Se puede observar cómo interactúan los dos actores estudiante-administrador con el sistema durante el caso de uso Pruebas Saber Pro (***Figura 6.1, 6.2, 6.3***).

Figura 6.1: Diagrama de Secuencias Pruebas Saber Pro StarUML



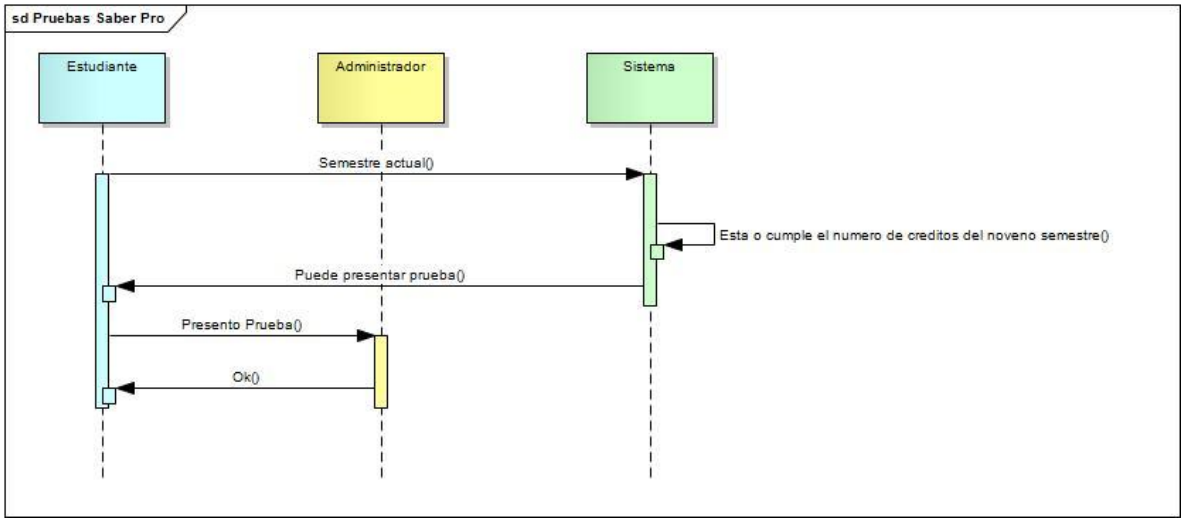
Fuente: Autores

Figura 6.2: Diagrama de Secuencias Pruebas Saber Pro Poseidon For UML



Fuente: Autores

Figura 6.3: Diagrama de Secuencias Pruebas Saber Pro Enterprise Architect

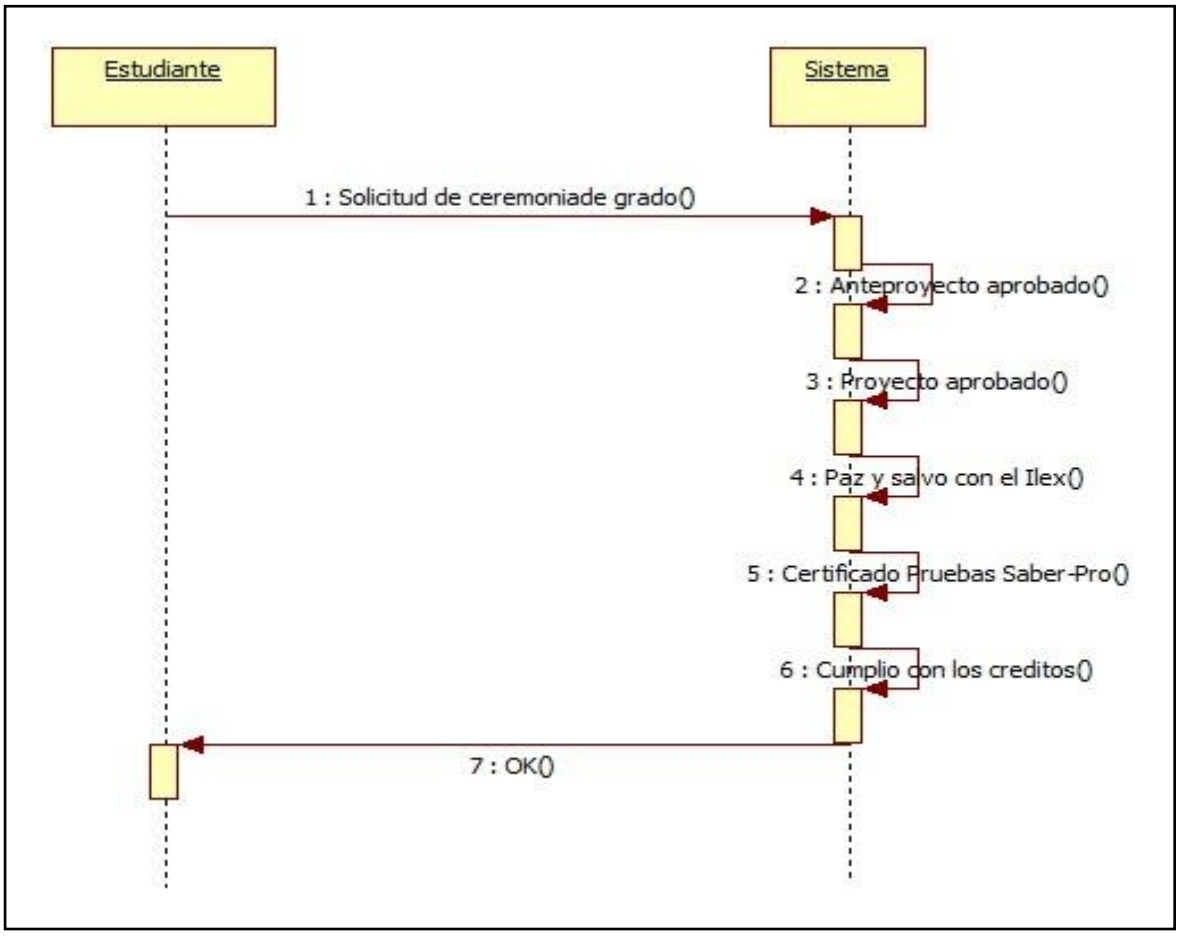


Fuente: Autores

Secuencias para el caso de uso Solicitar Grado

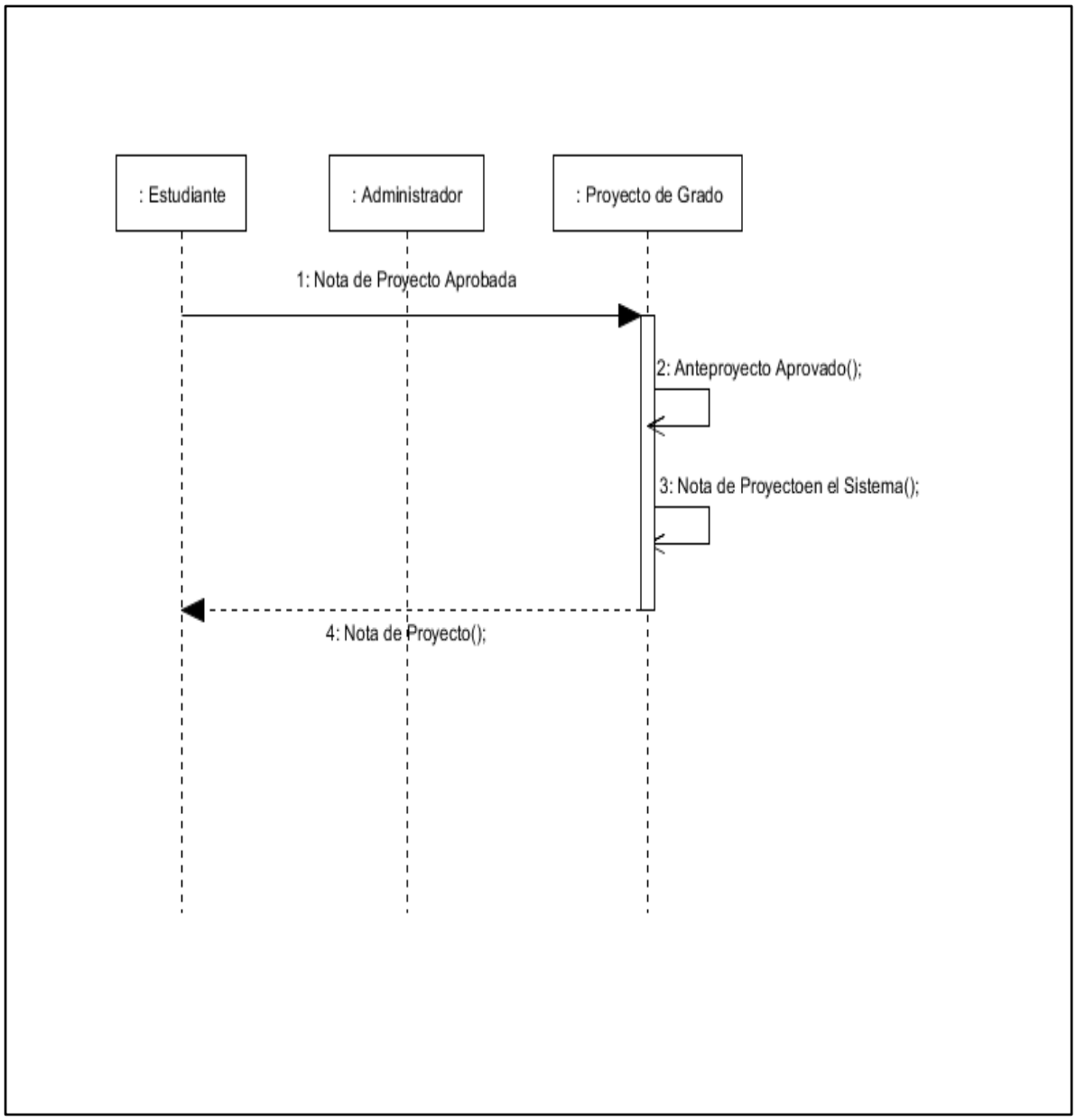
En este diagrama se puede observar la interacción final entre el estudiante y el sistema en el caso de uso Solicitar grado, el cual reúne la serie de pasos final en donde se engloban los demás casos de uso, para que este se cumpla (***Figura 7.1, 7.2, 7.3***).

Figura 7.1:Diagrama de Secuencias Solicitar Grado StarUML



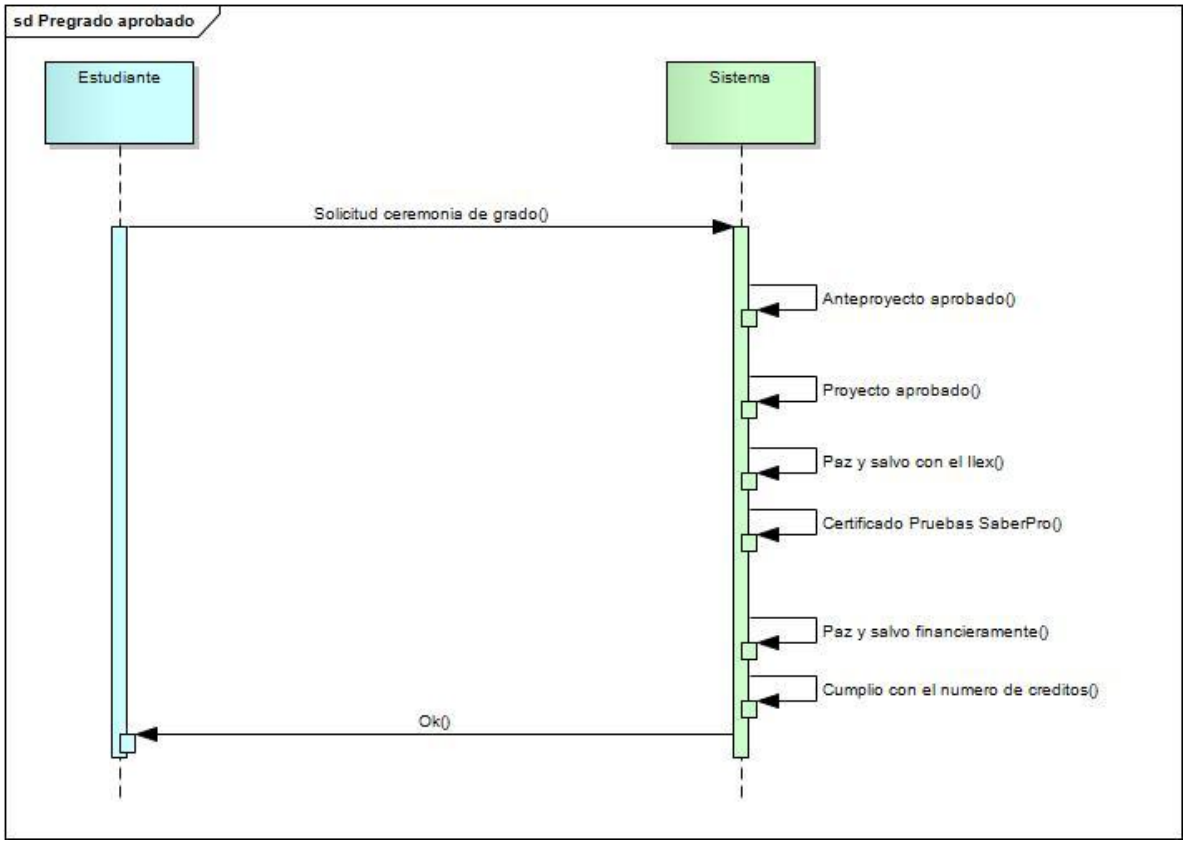
Fuente: Autores

Figura 7.2: Diagrama de Secuencias Solicitar Grado Poseidon For UML



Fuente: Autores

Figura 7.3: Diagrama de Secuencias Solicitar Grado Enterprise Architect



Fuente: Autores

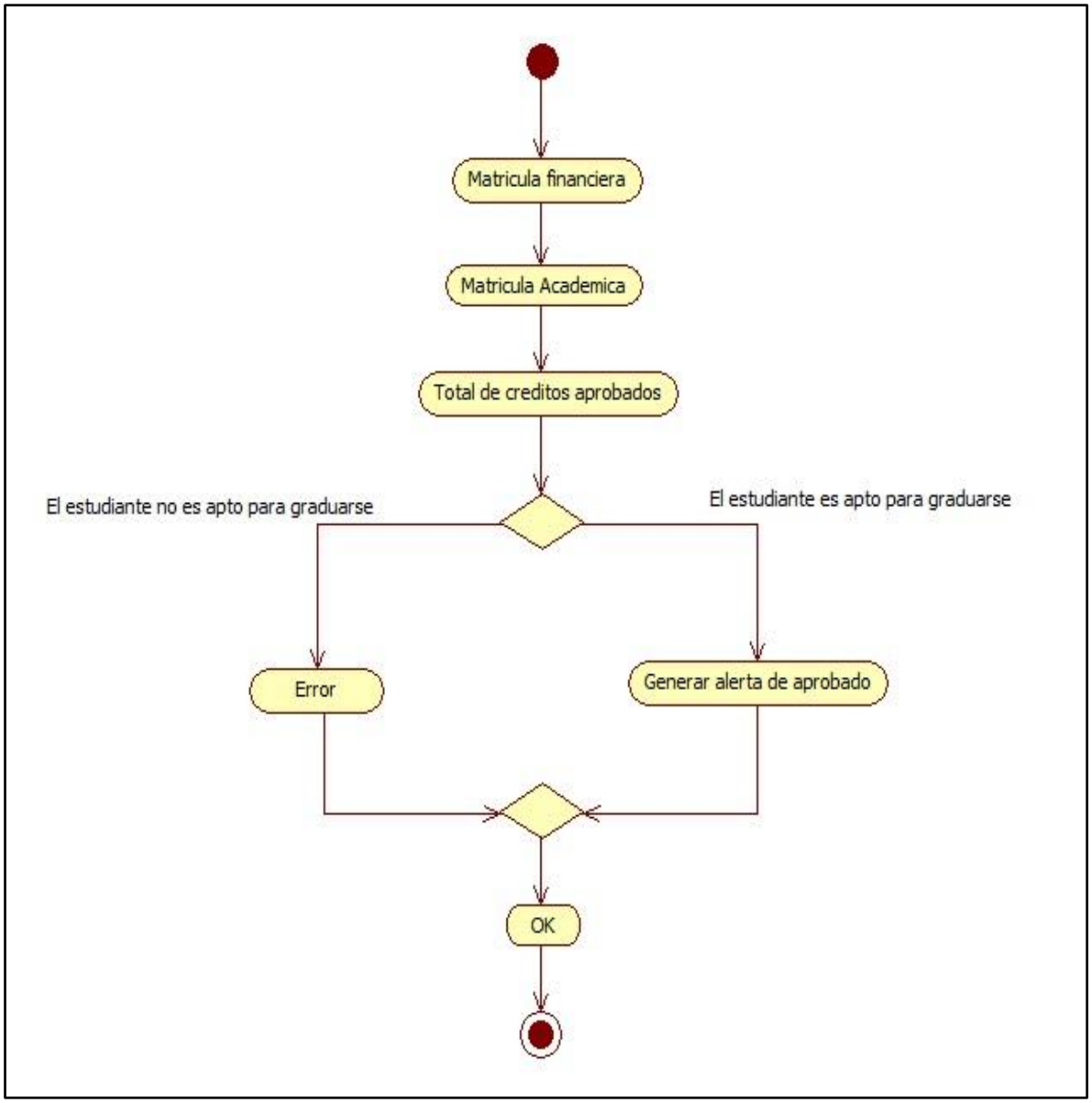
4.6 DIAGRAMA DE ACTIVIDADES

Son aquellos en los cual se muestra la serie de pasos para cada caso de uso. Son similares a los diagramas de flujo pero cabe aclarar que no son lo mismo. Estos ayudan a entender al equipo de desarrollo como se utiliza el sistema y cómo reacciona frente a diferentes eventos que se pueden presentar durante la ejecución.

Diagrama de Asignaturas

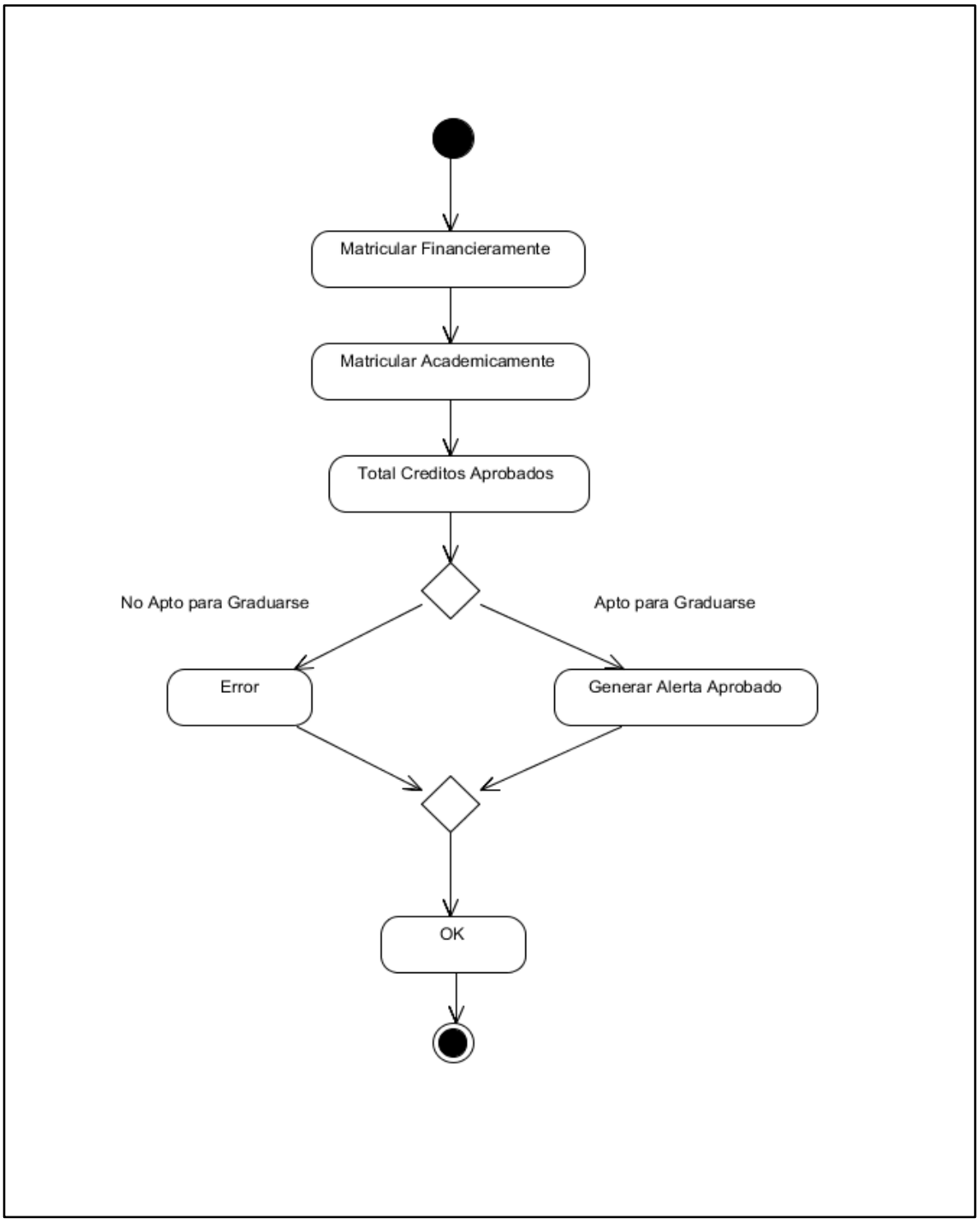
Mediante este diagrama se describe la serie de pasos que debe seguir el caso de uso Aprobar asignaturas (***Figura 8.1, 8.2, 8.3***).

Figura 8.1: Diagrama de Actividades Asignaturas StarUML



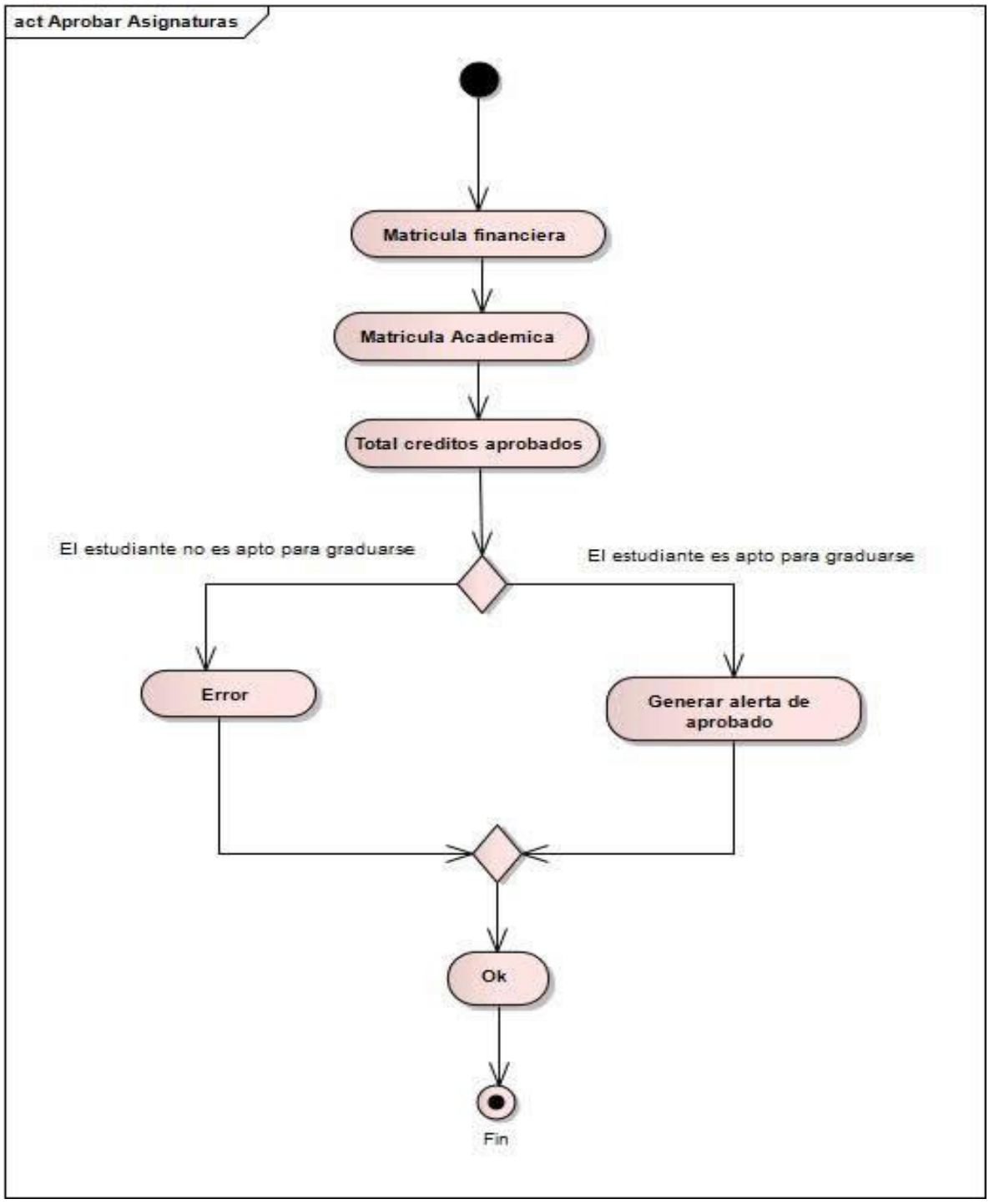
Fuente: Autores

Figura 8.2: Diagrama de Actividades Asignaturas Poseidon For UML



Fuente: Autores

Figura 8.3: Diagrama de Actividades Asignaturas Enterprise Architect

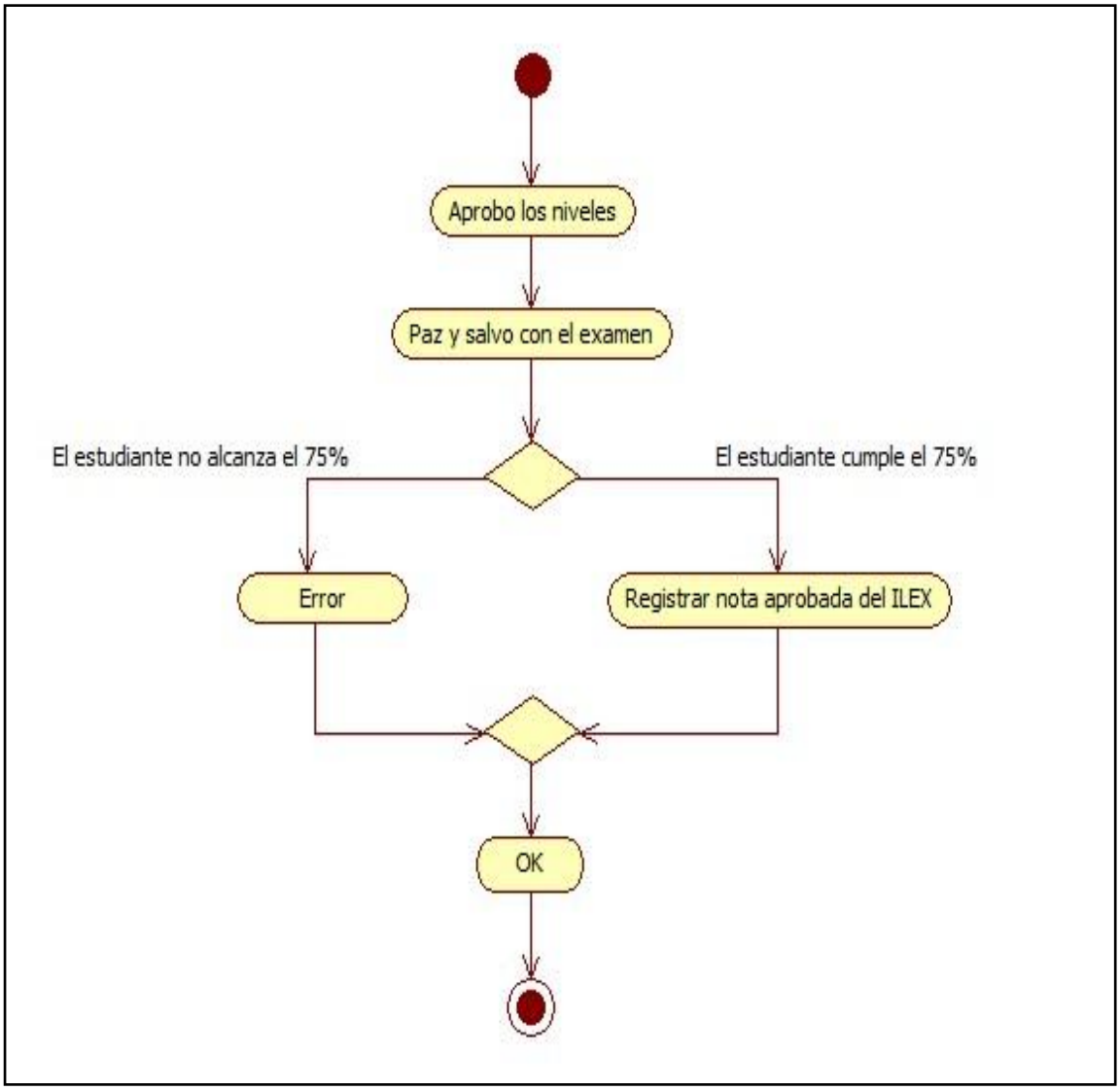


Fuente: Autores

Diagrama de actividad ILEX

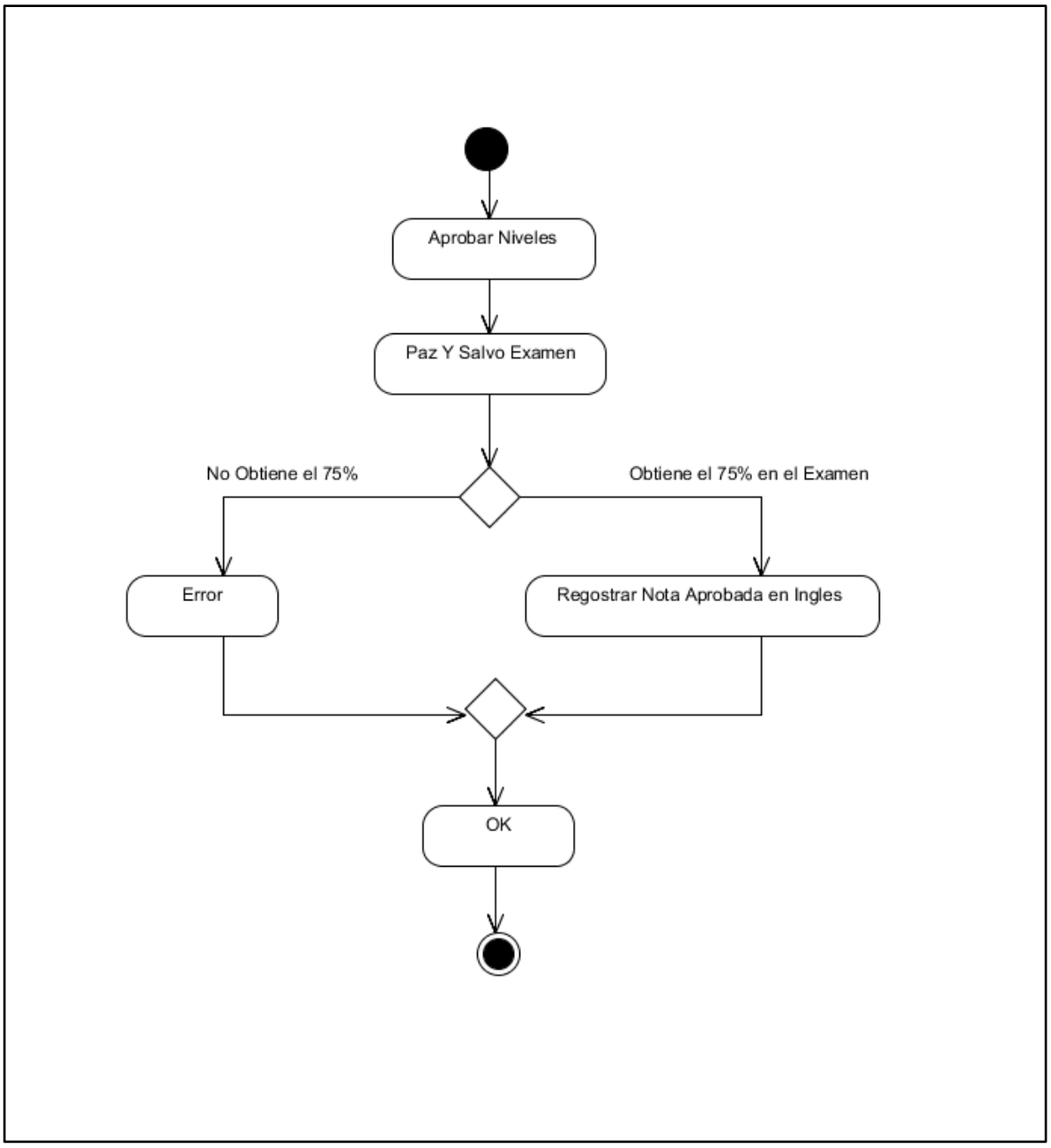
En este se observar la secuencia de pasos que debe seguir el caso de uso de Examen de Inglés (**Figura 9.1, 9.2, 9.3**).

Figura 9.1: Diagrama de Actividades ILEX StarUML



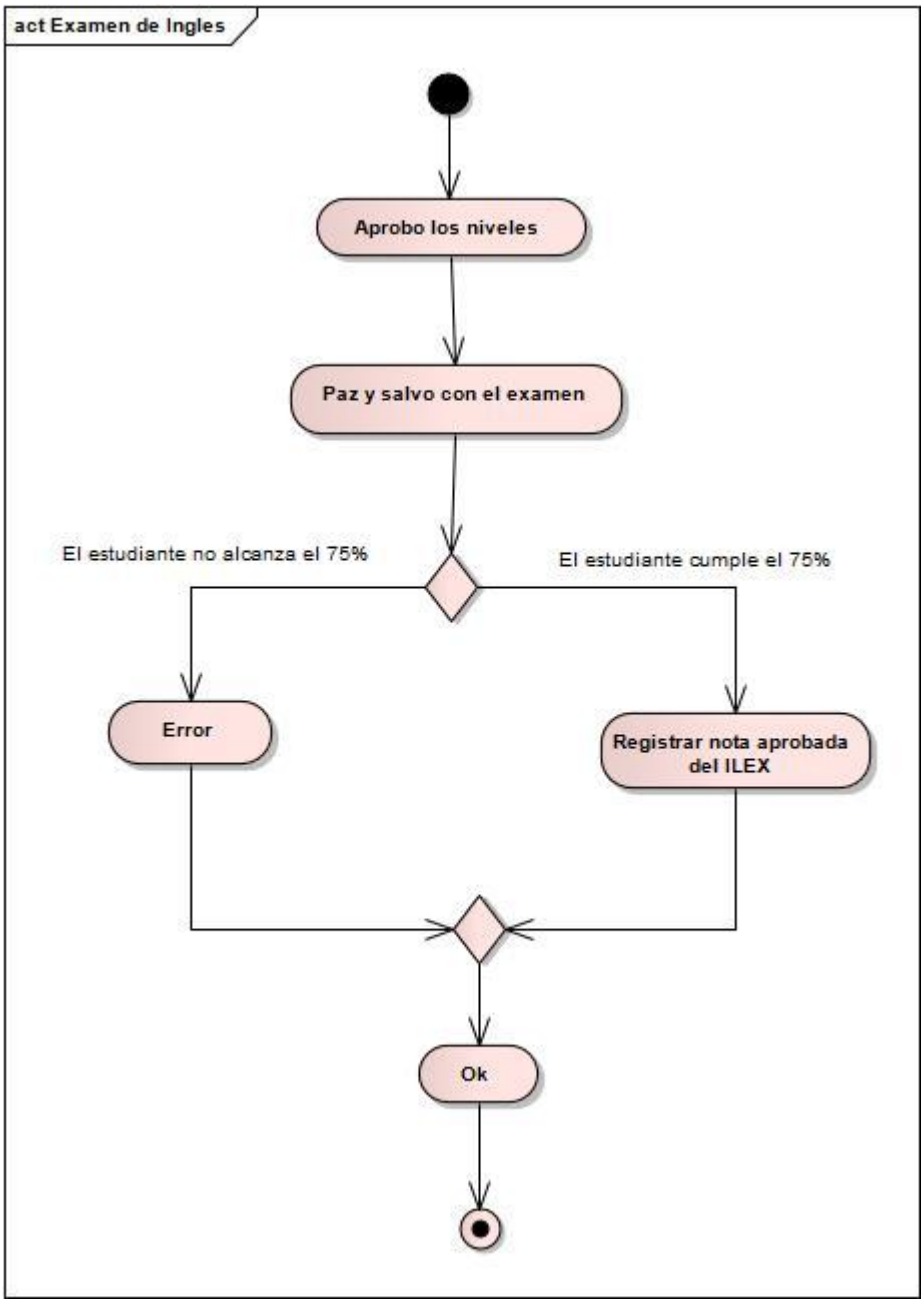
Fuente: Autores

Figura 9.2: Diagrama de Actividades INGLES Poseidon For UML



Fuente: Autores

Figura 9.3: Diagrama de Actividades ILEX Enterprise Architect

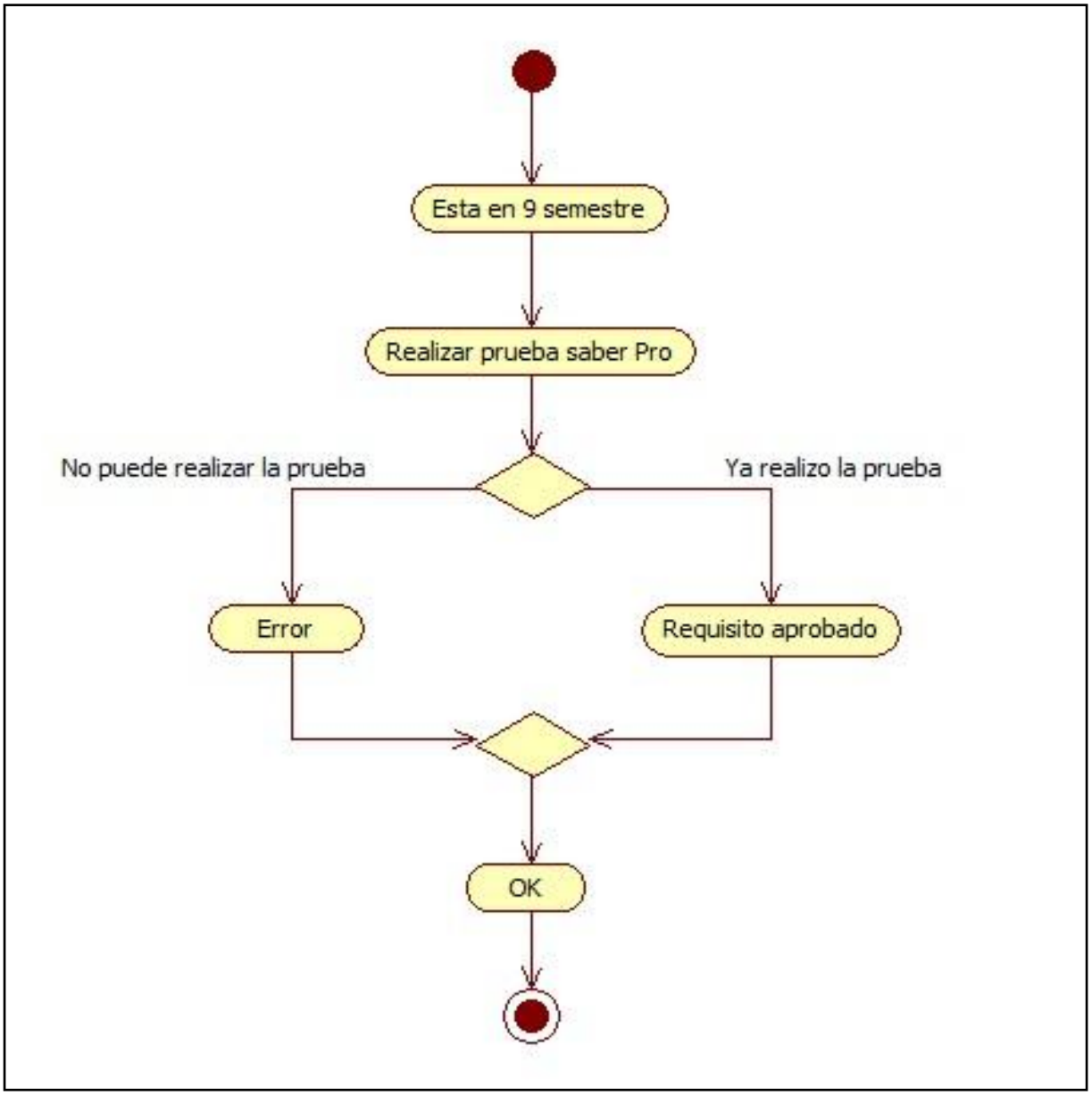


Fuente: Autores

Diagrama de actividad SABER PRO

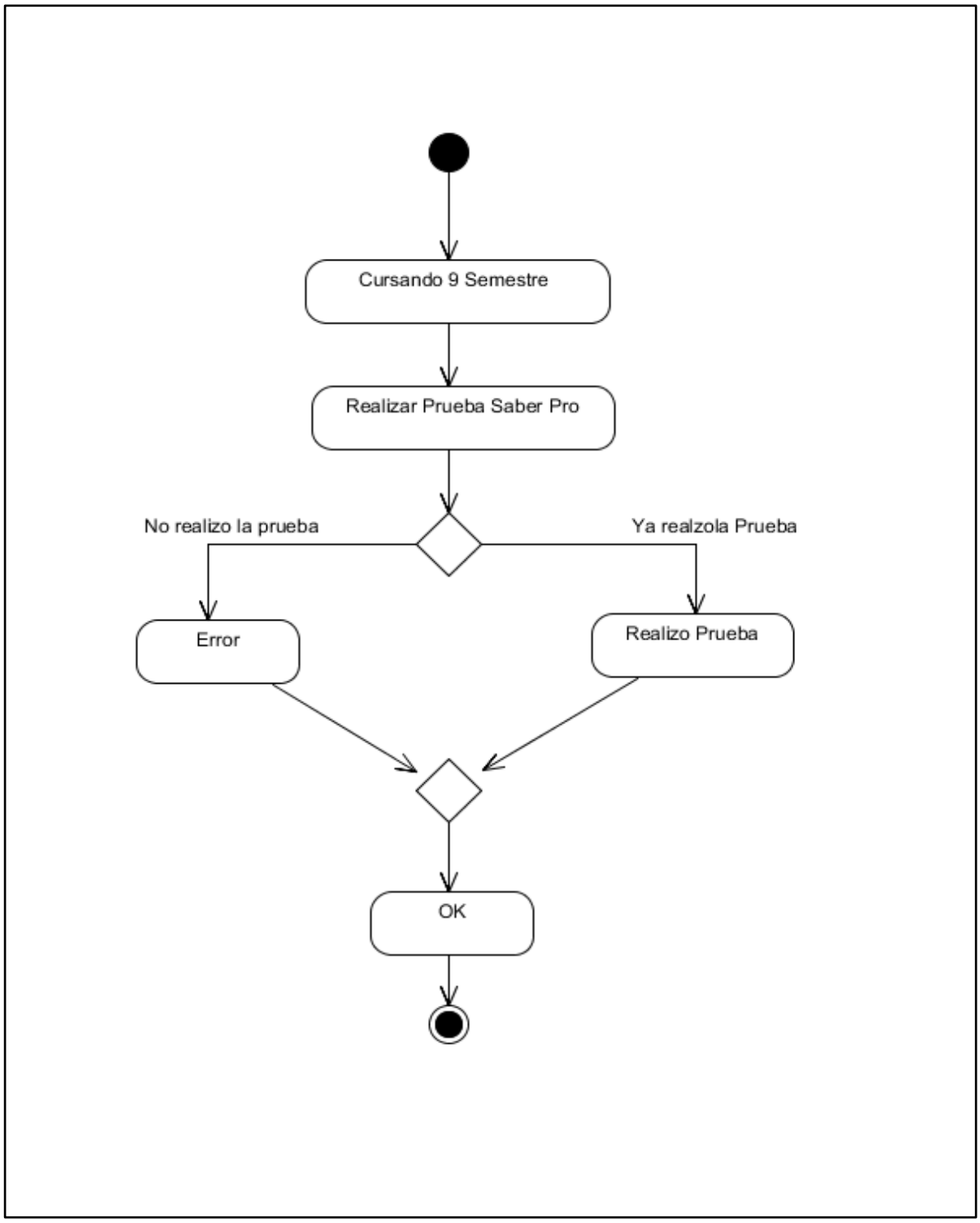
Al igual que en los otros diagramas, se muestran los pasos del caso de uso de las Pruebas Saber-PRO (***Figura 10.1, 10.2, 10.3***).

Figura 10.1: Diagrama de Actividades Pruebas Saber PRO STARUML



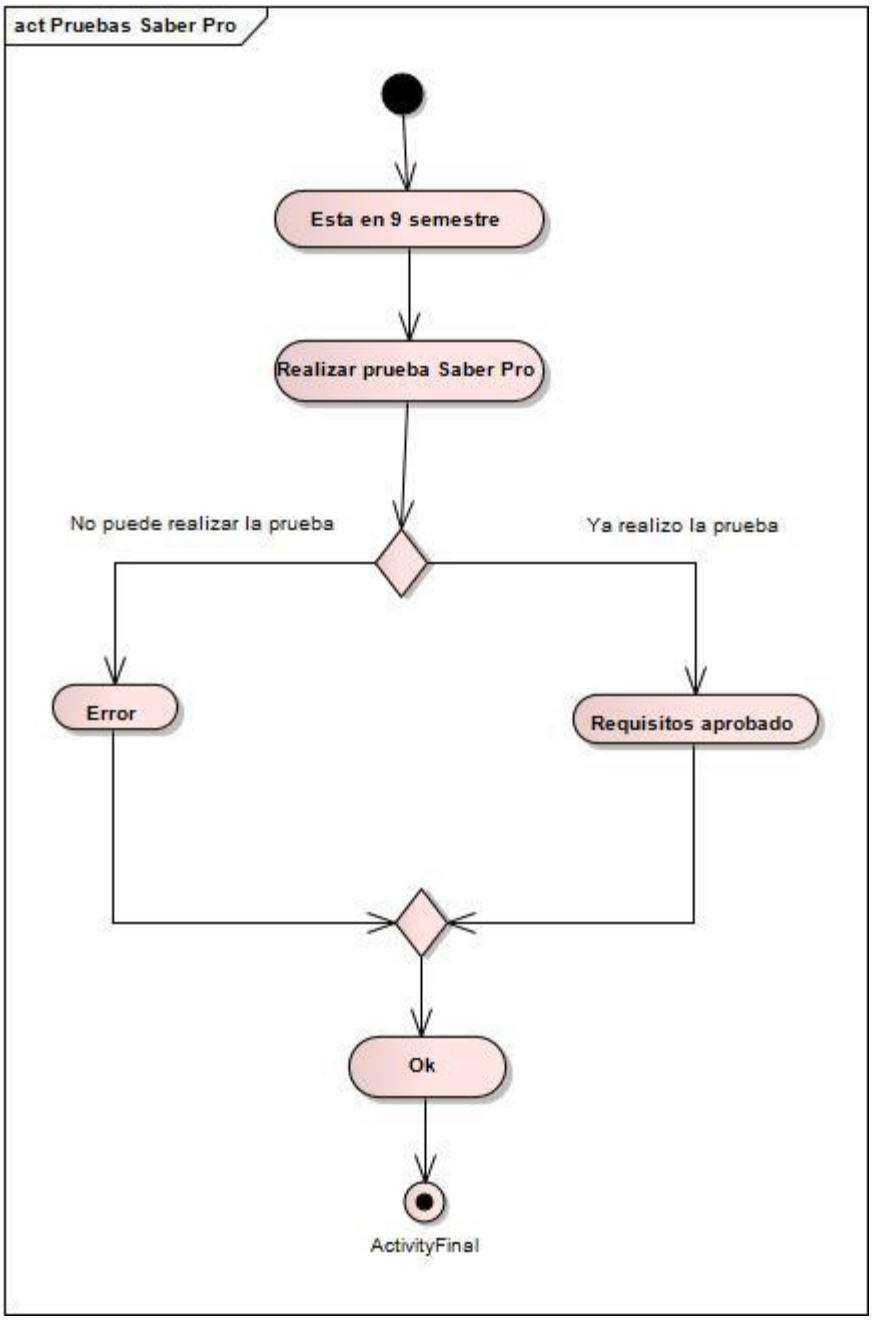
Fuente: Autores

Figura 10.2: Diagrama de Actividades Pruebas Saber PRO Poseidon For UML



Fuente: Autores

Figura 10.3: Diagrama de Actividades Pruebas Saber PRO

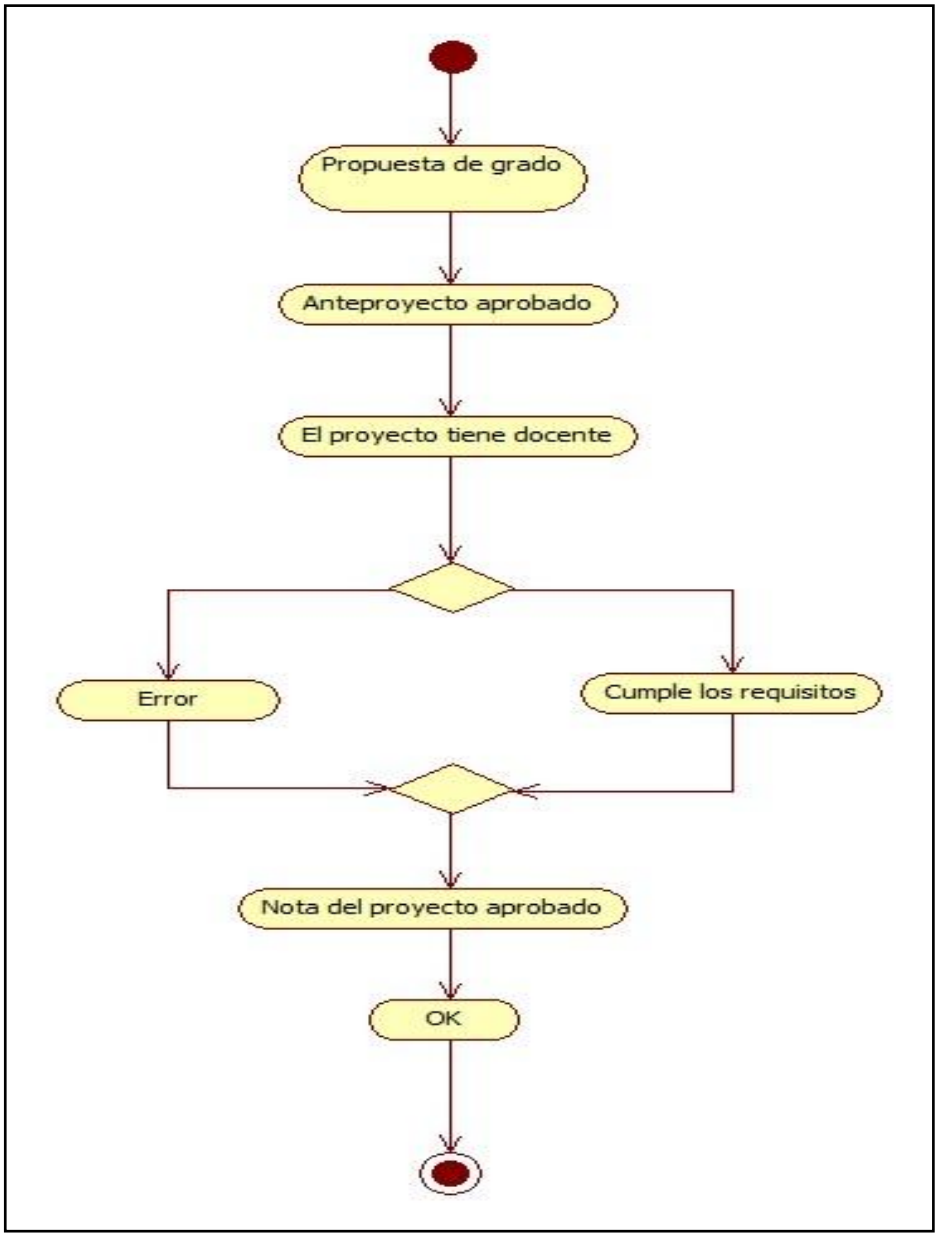


Fuente: Autores

Diagrama de actividad Proyecto

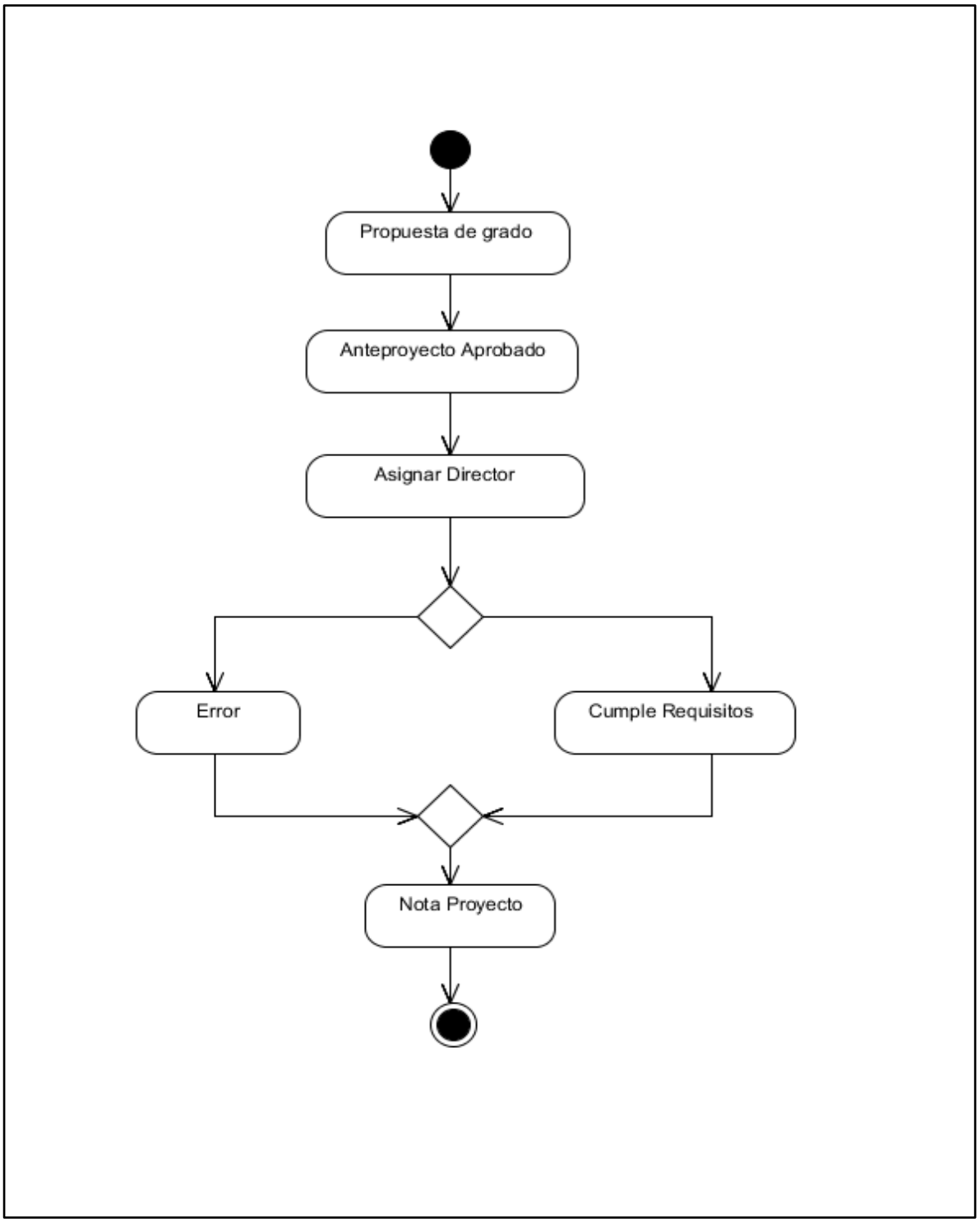
En este diagrama se ven los pasos que se siguen en el caso de uso Proyecto de grado (***Figura 11.1, 11.2, 11.3***).

Figura 11.1: Diagrama de Actividades Proyecto StarUML



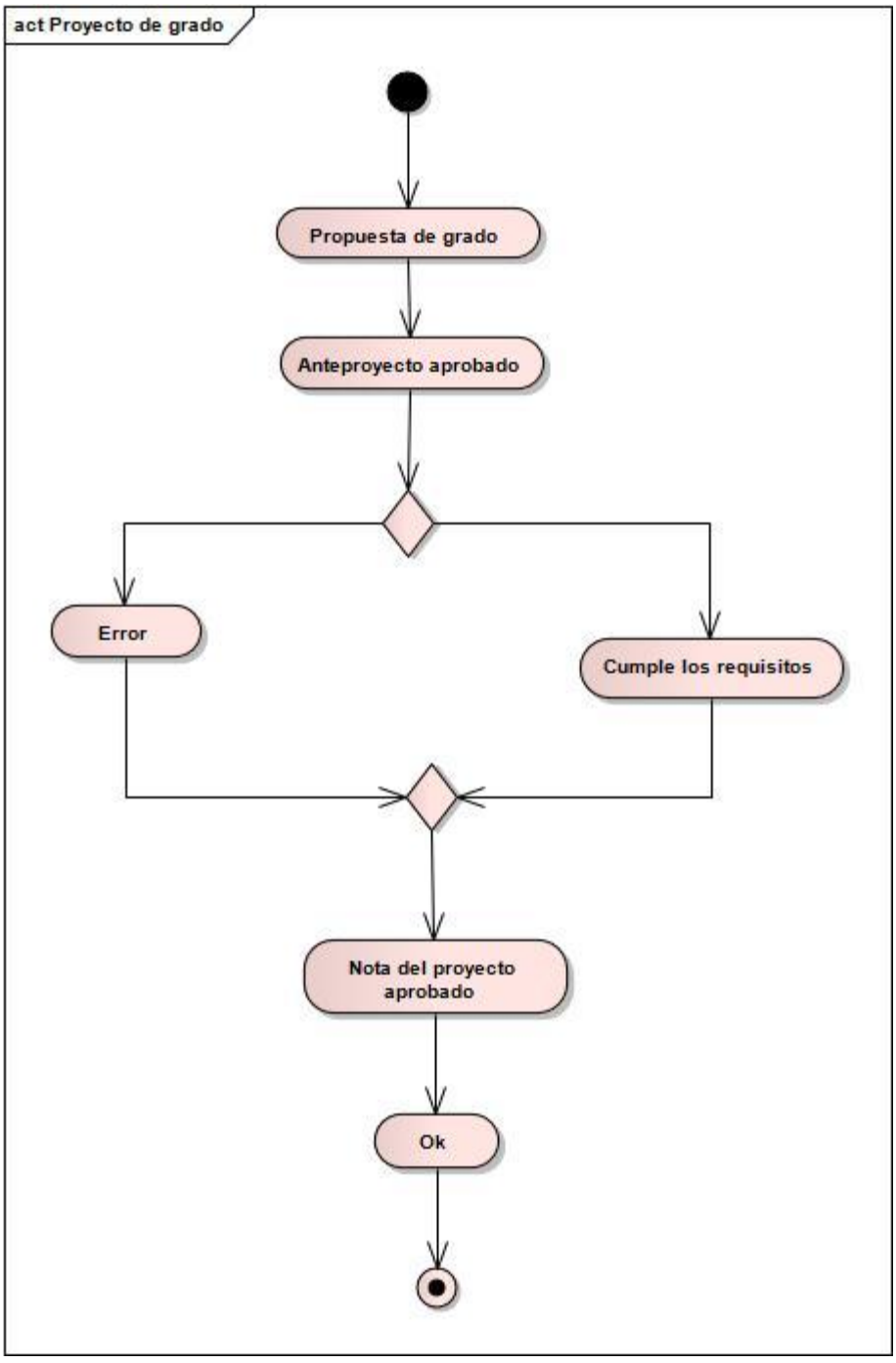
Fuente: Autores

Figura 11.2: Diagrama de Actividades Proyecto Poseidon For UML



Fuente: Autores

Figura 11.3: Diagrama de Actividades Proyecto Enterprise Architect

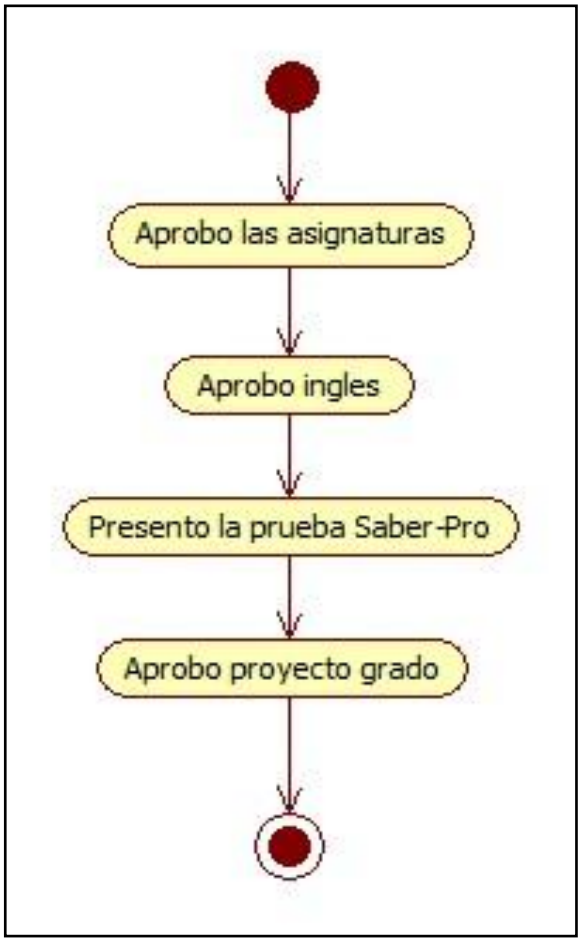


Fuente: Autores

Diagrama de actividad Pregrado

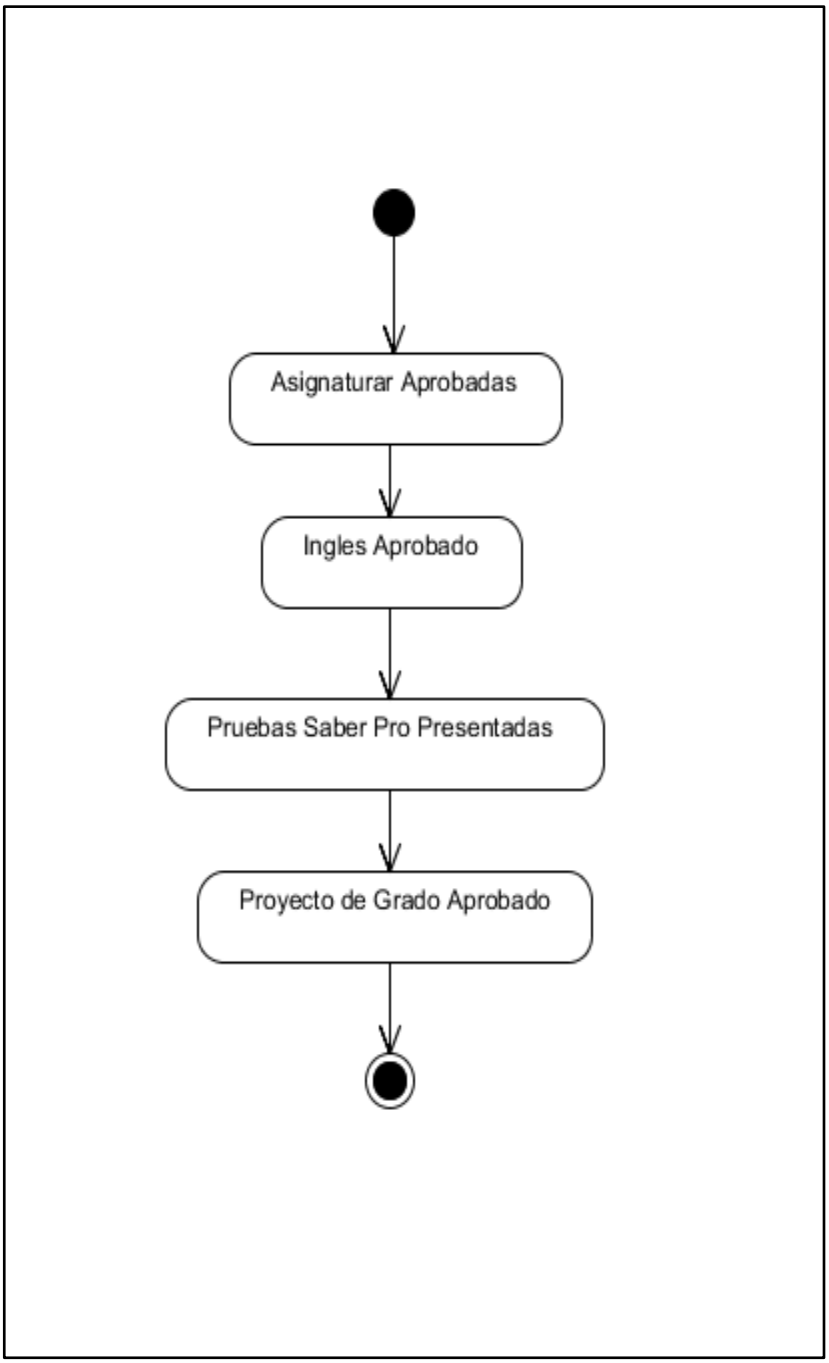
En él se observan la serie de pasos del caso de uso final, en el cual se tienen en cuenta los demás casos de uso. Este es el caso de uso de pregrado, el paso final para poder optar al título profesional (***Figura 12.1, 12.2, 12.3***).

Figura 12.1: Diagrama de Actividades Pregrado



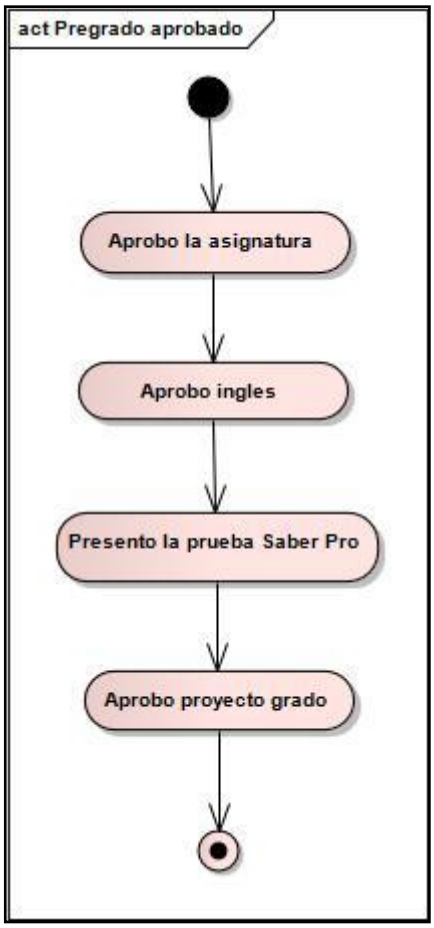
Fuente: Autores

Figura 12.2: Diagrama de Actividades Pregrado



Fuente: Autores

Figura 12.3: Diagrama de Actividades Pregrado Enterprise Architect



Fuente: Autores

5. MODELO DE EVALUACIÓN DE LAS HERRAMIENTAS

El modelo de evaluación de las herramientas CASE: Enterprise Architect, StarUML y Poseidon For UML, para modelamiento de diagramas UML, está basado en métricas para el desarrollo de software y son apoyadas en estándares de calidad de la ISO 9126. Para comprender porque evaluar por medio de métricas es necesario hacer referencia a la definición de métricas, los tipos de métricas que existen y su aporte en el ambiente evaluativo.

La ISO 9126 proporciona modelos de calidad para software específico, usualmente es desarrollada como una cuantificación de 4 pasos:

1. Identificación de la calidad de los requerimientos.
2. Identificación del contexto de interpretación, el cual es: selección de valores de referencia. Al igual que la determinación de los objetivos específicos para un contexto particular
3. Uso derivado de las medidas para los datos realizados en el paso uno.
4. Comparación de los resultados del paso tres con los objetivos del paso dos, para tomar una decisión basados en la información obtenida y cualquier otro tipo de información relevante que permite tomar una decisión[8].

[8] ISO 9126;[Pag 208-209]
http://books.google.com.co/books?id=rvF1nsgwa54C&pg=PA205&dq=ISO+9126&hl=es&sa=X&ei=lONiT7n9G87ugge0lJDnAg&ved=0CDUQ6AEwAA#v=onepage&q=ISO%209126&f=false

5.1 MÉTRICAS

Dicho anteriormente se definen las métricas para este caso como todas aquellas medidas que están directamente relacionadas con el desarrollo del software. En ellas se encuentran varios tipos, como son:

- Las Métricas Técnicas se centran en las características de software y miden la estructura del sistema, el cómo está hecho.

- Las Métricas de Calidad son aquellas que proporcionan una indicación de cómo se ajusta el software a los requisitos implícitos y explícitos del cliente.

- Métricas de Productividad se centran en el rendimiento del proceso de la ingeniería del software.

- Las Métricas Orientadas a la Persona proporcionan medidas e información sobre la forma en que la gente desarrolla el software de computadoras y en especial el punto de vista humano de la efectividad de las herramientas y métodos.
- Las Métricas Orientadas Al Tamaño consisten en conocer en qué tiempo se va a terminar el software y cuántas personas se van a necesitar. Son medidas directas al software y el proceso por el cual se desarrolla si una organización de software mantiene registros sencillos[9].

En el siguiente diagrama (***figura 13***) se muestran las métricas que se pretenden usar para la evaluación de las herramientas, en este estudio; por medio de las cuales es posible estimar en qué nivel se cumple con las métricas seleccionadas.

[9] Gonzales Doria. Heidi. Las métricas de software y su uso en la región. Cholula, Puebla. México. Mayo 7 de 2001. Capitulo 3. http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/gonzalez_d_h/ [En Línea] [Citado Marzo de 2012]

Figura 13: Métrica de Evaluación

Fuente: Autores

Para comprender de manera generalizada como es que las herramientas pueden ser valoradas con estas métricas, se realiza una descripción generalizada de las métricas que fueron elegidas.

5.1.1 FIABILIDAD

En los sistemas de información existe tanto fiabilidad en el hardware como en el software. En el caso de las herramientas CASE, se tiene en cuenta el enfoque

estrictamente en la fiabilidad del software, es decir, en la herramienta, sin desconocer la importancia del hardware, que será relevante en este estudio, y no se cierran las puertas para estudios posteriores. En términos estadísticos, la fiabilidad es definida como una probabilidad, el número de veces que el programa puede operar libre de fallos durante un tiempo y entorno determinado, es decir, la probabilidad de hasta dónde puede el programa llevar a cabo su función con la exactitud requerida.

En la calidad del software cualquier tipo de falla sin importar su significancia, afecta la sincronía con respecto a los requisitos y requerimientos del software; la fiabilidad es una de las métricas más influyentes en el momento de medir o hablar de calidad.

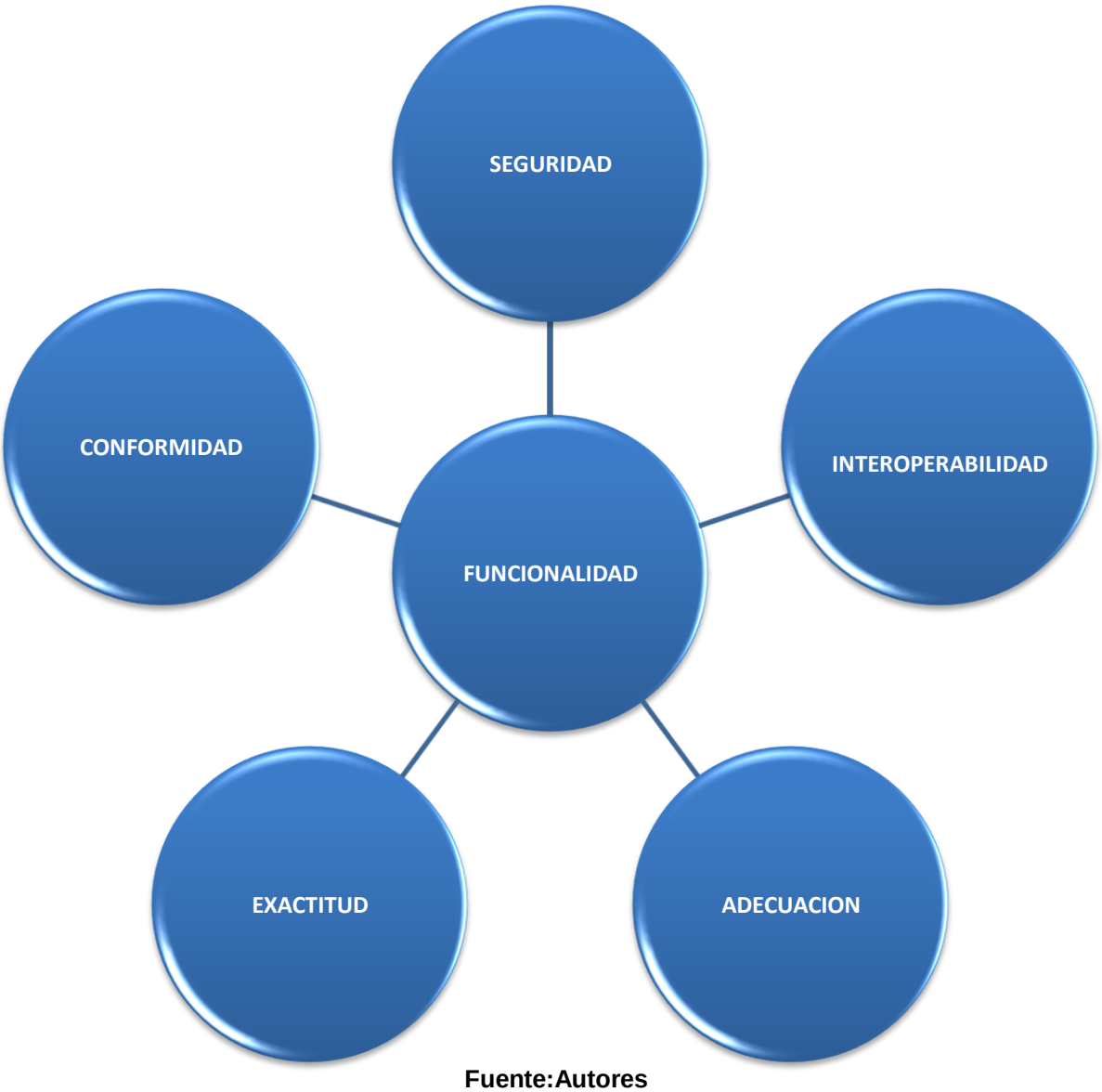
Figura 14: Fiabilidad

Fuente: Autores

5.1.2 FUNCIONALIDAD

Cuando se habla de funcionalidad estrictamente se refiere a que la herramienta o software cumpla adecuadamente con las necesidades para las cuales fue diseñada o es usada. En el estándar de calidad ISO 9126 se establece un conjunto de atributos que facilitan la calificación. Estos son:

Figura15: Funcionalidad



Fuente:Autores

- **Adecuación.** Se enfoca en evaluar si el software cuenta con un conjunto de funciones apropiadas para efectuar las tareas que fueron especificadas en su definición[10].

[10]Pag 2/ funcionalidad http://www.revistaupiicsa.20m.com/Emilia/RevEneAbr04/Antonieta1.pdf

- **Exactitud.** Este atributo permite evaluar si el software presenta resultados o efectos acordes a las necesidades para las cuales fue creado.

- **Interoperabilidad.** Permite evaluar la habilidad del software de interactuar con otros sistemas previamente especificados.

- **Conformidad.** Evalúa si el software se adhiere a estándares, convenciones o regulaciones en leyes y prescripciones similares.

- **Seguridad.** Se refiere a la habilidad de prevenir el acceso no autorizado, ya sea accidental o premeditado, a los programas y datos[11].

5.1.3 MANTENIBILIDAD

El IEEE (19990) define mantenibilidad como: “La facilidad con la que un sistema o componente software puede ser modificado para corregir fallos, mejorar su funcionamiento u otros atributos o adaptarse a cambios en el entorno”[12].

Es decir que se refiere a los atributos que permiten medir el esfuerzo necesario para realizar modificaciones al software, ya sea por la corrección de errores o por el incremento de funcionalidad.
En este caso de evaluación, los cuatro factores que establece el estándar de la ISO 9126 para evaluar esta métrica son la base que se tiene en cuenta. Estos son:

[11]Pag 2/ funcionalidad http://www.revistaupiicsa.20m.com/Emilia/RevEneAbr04/Antonieta1.pdf
[12]Definición de la ieee para mantenibilidadhttp://cnx.org/content/m17452/latest/

Figura 16: Mantenibilidad

Fuente: Autores

Capacidad de Análisis. Relativo al esfuerzo necesario para diagnosticar las deficiencias o causas de fallas, o para identificar las partes que deberán ser modificadas.

Capacidad de Modificación. Mide el esfuerzo necesario para modificar aspectos del software, remover fallas o adaptar el software para que funcione en un ambiente diferente.

Estabilidad. Permite evaluar los riesgos de efectos inesperados debidos a las modificaciones realizadas al software.

Facilidad de Prueba. Se refiere al esfuerzo necesario para validar el software una vez que fue modificado.

5.1.4 PORTABILIDAD

La portabilidad se refiere a la habilidad del software de ser transferido de un ambiente a otro, la idea es que sin importar el sistema operativo, software o versión, la herramienta cumpla las mismas funciones, al igual que los documentos elaborados en ella sobre la misma herramienta no sufra trasformaciones o fallos al ser abierto o usado en la otra herramienta y considera los siguientes aspectos:

Figura 17: Portabilidad

Fuente: Autores

Adaptabilidad. Evalúa la oportunidad de adaptar el software a diferentes ambientes sin necesidad de aplicarle modificaciones.

Facilidad de Instalación. Es el esfuerzo necesario para instalar el software en un ambiente determinado.

Capacidad de Reemplazo. Se refiere a la oportunidad y el esfuerzo usado en sustituir el software por otro producto con funciones similares.

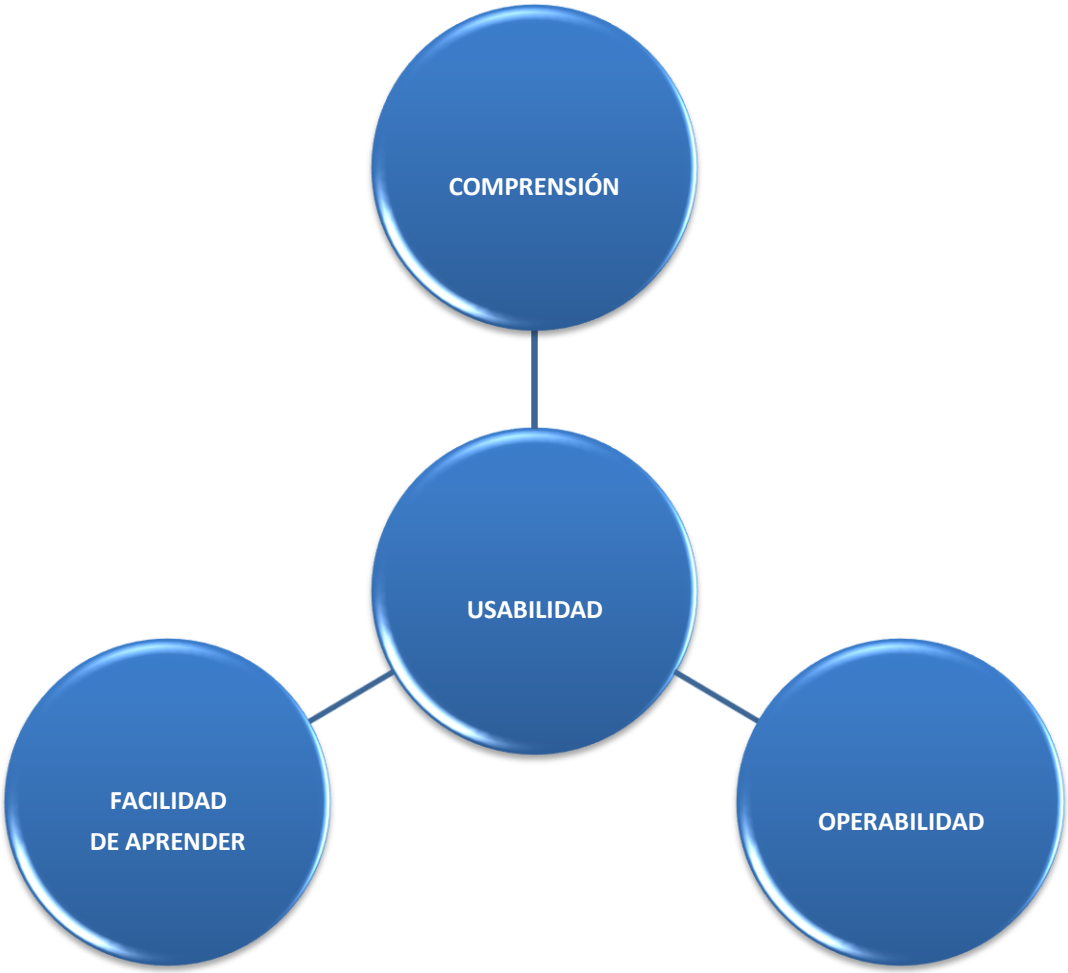
5.1.5 USABILIDAD

Si un proyecto no tiene calidad no es productivo en el mercado. Por tal razón, la usabilidad juega un papel importante debido a que va ligada a la calidad del software. Basados en el estándar ISO 9126 se encuentra que la usabilidad es definida como "el grado en el que un producto puede ser utilizado por usuarios específicos para conseguir objetivos específicos con efectividad, eficiencia y satisfacción en un determinado contexto de uso". Lo que significa que este estándar relaciona la herramienta con el usuario para alcanzar su satisfacción. El sistema debe estar construido para alcanzar la usabilidad medida en niveles cuantitativos, como se hace en el trascurso de este proyecto.

Al igual que el estándar anterior, la norma ISO 9126 la define como "Un conjunto de atributos que permiten evaluar el es fuerzo necesario que deberá invertir el usuario para utilizar el sistema"[13]. Para ser evaluado lo dividen en tres atributos, los cuales se tienen en cuenta en el momento de medir cada una de las tres herramientas CASE.

[13] PAG 2/ USABILIDAD http://www.revistaupiicsa.20m.com/Emilia/RevEneAbr04/Antonieta1.pdf

Figura 18: Usabilidad



Fuente: Autores

- **Comprensión.** Se refiere al esfuerzo requerido por los usuarios para reconocer la estructura lógica del sistema y los conceptos relativos a la aplicación del software.

- **Facilidad de Aprender.** Establece atributos del software relativos al esfuerzo que los usuarios deben hacer para aprender a usar la aplicación.

- **Operabilidad.** Agrupa los conceptos que evalúan la operación y el control del sistema.

5.1.6 EFICIENCIA

Cuando se habla de desarrollos de software una de las métricas más comunes que se mencionan es la eficiencia, pero inmediatamente se presenta una dificultad la cual es confundir este término con la eficacia; la eficiencia es la relación entre el número de recursos informáticos usados y el código necesario para realizar alguna función antes, durante y después del proyecto, es decir, para este caso, la capacidad de la herramienta de hacer un buen uso de los recursos,- entendiéndose recursos como todo componente del hardware y software que sean necesarios para el funcionamiento y optimización del trabajo[14]-. Mientras que la eficacia es el nivel en el cual se consiguen las metas y objetivos del proyecto.

[14]Eficiencia http://informacion.wordpress.com/2006/06/06/%C2%BFque-son-los-recursos-informaticos/

Figura 19: ASPECTOS PARA MEDIR LA EFICIENCIA

Fuente: Autores

Comportamiento con Respecto al Tiempo. Atributos del software relativos a los tiempos de respuesta y de procesamiento de los datos.

Comportamiento con Respecto a Recursos. Atributos del software relativos a la cantidad de recursos usados y la duración de su uso en la realización de sus funciones.

5.1.7 INTEGRIDAD

Como su nombre lo indica, y más cuando se habla de ingeniería del software, la integridad es el estado en que los datos están sin modificaciones no autorizadas.

La integridad puede ser violada por un usuario sin quererlo, por error o con pre limitación, elimina o altera la información del proyecto o de la herramienta. En esta métrica también se deben tener en cuenta las medidas que facilitan o proporcionan la herramienta para evitar esta pérdida accidental.

Figura 20: Integridad

Fuente: Autores

6. CRITERIOS Y EVALUACION

Es oportuno aclarar que todos los criterios acá expuestos son desde el puntos de vistas como usuarios de la herramienta, como se menciona anteriormente el criterio se fundamenta en normas de calidad y el conocimiento adquirido a lo largo de la vida académica, este documento es un apoyo investigativo, y deja abierta la opinión de las personas que lo usen o manejen un criterio desde su punto de vista ya que este puede ser muy subjetivo, acorde a lo dicho anteriormente se pasa a explicar cómo se evaluó cada métrica y los subgrupo de componentes, y los criterios de evaluación que se observa en cada una de las herramientas.

6.1 FIABILIDAD

Como se menciona anteriormente en esta métrica la evaluación se enfoca en el numero de fallos que se perciben en la herramientas durante el tiempo que fue usada para el desarrollo de los diagramas.

Los tipos de fallas:

- De instalación
- De soporte
- De procesamiento
- De diseño

Tabla 6. CRITERIOS DE EVALUACIÓN

CRITERIOS DE EVALUACION	
VALOR	**DEFINICION**
1	Muy deficiente
2	Deficiente
3	Regular
4	Buena
5	Excelente

Fuente: Autores

Tabla 7. FIABILIDAD

HERAMIENTAS		
STARUML	**CALIFICACION**	**4**
	OBSERVACIONES	• En soporte se considera que es regular, por ser una herramienta libre el usuario debe ser autosuficiente en el soporte y esto puede generar obstáculos al momento de buscar la solución, porque posiblemente no sea la más acorde. • En cuanto al diseño, no es claro, porque en su parte grafica, concretamente las barras de herramientas carecen de claridad en sus componentes.
POSEIDON FOR UML	**CALIFICACION**	**5**
	OBSERVACIONES	• Se considera a Poseidón For UML una herramienta poderosa para el modelamiento de diagramas UML, en aspectos de soporte posee una gran comunidad a la disposición, además Gentleware la empresa encargada del desarrollo y distribución ofrece buen soporte por el hecho de ser una herramienta licenciada. • La interfaz de usuario presenta facilidad y amabilidad con los usuarios, los repositorios están bien ubicados y las funciones o herramientas que ofrece son de fácil acceso y uso.
ENTERPRISE ARCHITECT	**CALIFICACION**	**5**
	OBSERVACIONES	• Una herramienta muy completa respecto al soporte, por ser licenciada ofrece a sus usuarios disponibilidad de soporte en cualquier falla. • El diseño es muy claro, aunque su nivel es un poco alto en cuanto a características permite ser explícito al momento de describir las clases, atributos, componentes y demás características que facilitaran la generación futura del código.

Fuente: Autores

6.2 FUNCIONALIDAD

Al igual que en las demás métricas en esta se analiza los diferentes aspectos frente a las propiedades descritas, en especial se detalla que la herramienta funcione como esta especificada, puesto que funcionalidad es lo que el producto puede hacer.

Para la evaluación se emplean algunas clasificaciones basadas en normas de calidad, para cada una se toman puntos de vistas enfocados, en la adecuación se observa si la herramienta cuenta con las funciones para efectuar las tareas que se requieren y en especial las que especifica el proveedor, las cuales se describen en el capítulo 2, en términos del léxico común “lo que prometen”.

En cuanto a exactitud se mira que tan acorde es con el objetivo para el cual fue creada la herramienta.

En cuanto a interoperabilidad se fija específicamente en que tan hábil será o es para actuar con otro software.

En la conformidad específicamente la evaluación se centra en observar si se adapta al estándar descrito por UML.

Y en seguridad la evaluación radica en que tanto acceso permite a sus datos.

Tabla8. CRITERIOS DE EVALUACIÓN FUNCIONALIDAD

CRITERIOS DE EVALUACION	
VALOR	**DEFINICION**
1	Nunca
2	Casi nunca
3	Algunas veces
4	Casi siempre
5	Siempre

Fuente: Autores

Tabla 9. EVALUACION FUNCIONALIDAD

HERAMIENTAS		ADECUACION	EXACTITUD	INTEROPERABILIDAD	SEGURIDAD	CONFORMIDAD
STARUML	CALIFICACION	5	5	4	3	5
	OBSERVACIONES	Tiene las herramientas necesarias para las funciones pero no posee una estructura organizada y clara para el uso de ella.	Cumple con lo prometido en la descripción proporcionada por el desarrollador en su página oficial.	La herramienta está en capacidad de intercambiar información con diferentes aplicaciones y software, un ejemplo claro es la generación de código en diferentes lenguajes y la exportación de diagramas en xml el cual es un metalenguaje.	La Seguridad en este caso se toma como la capacidad de la herramienta de garantizar la integridad de los datos a los que el usuario opera, haciendo énfasis en el cómo se puede recuperar de un error, StarUML no genera autoguardados periódicamente para no perder información, aunque mientras se está trabajando sobre la herramienta no se detectan perdidas de datos entre una operación y otra.	Se le da la mayor calificación porque realmente se adapta a los estándares de UML pero aunque se es redundante en la aclaración, se considera oportuno recalcar que es desordenada y deficientes con respecto a la claridad de sus componentes gráficos para la construcción de los elementos.
POSEIDON FOR UML	CALIFICACION	5	5	4	5	3
	OBSERVACIONES	Permite a los usuarios entender fácilmente el uso de cada una de sus herramientas. Ahorra tiempo a la hora de realizar diagramas derivados de otros diagramas	Gentleware ofrece una herramienta fácil de usar y fácil de instalar, esto es lo que obtienen los usuarios al usar Poseidon For UML CE.	Genera código para java y HTML , soporta gran parte de los formatos de imágenes, es compatible entre ediciones	Garantiza la integridad de los Datos que el usuario opera generando autoguardados cada cierto tiempo durante su ejecución	Cumple con los estándares de Modelamiento UML, ya que relaciona entre si los diagramas que en este se pueden editar, además es una herramienta intuitiva.

HERAMIENTAS		ADECUACION	EXACTITUD	INTEROPERABILIDAD	SEGURIDAD	CONFORMIDAD
ENTERPRISE ARCHITECT	**CALIFICACION**	**5**	**5**	**5**	**5**	**5**
	OBSERVACIONES	Muy completa, en su última versión fue mejorada su caja de herramientas y de lenguajes que soporta. Por sus múltiples idiomas da mayores posibilidades de entendimiento dependiendo el usuario y su parte visual en al que se muestran descripciones de cada elemento.	Muy acorde para su fin, cumple con lo prometido, es demasiado completa, se necesitaría un proyecto muy robusto para explotar sus funcionalidades al máximo.	Genera reportes con la información necesaria dependiendo el formato que el usuario requiere, provee generación de documentos y herramientas con el complemento editor de plantillas WYSIWYG. Produce versiones html, genera código en java, c#, .net, python y entre otros lenguajes de alto nivel. Realiza ingeniería inversa en diferentes lenguajes.	Garantiza la integridad de los datos mientras el usuario tiene en uso la herramienta, sin importar el motivo por el cual cierre la herramienta, este genera copias constantes, lo cual no permite perdidas de información	Se adapta totalmente a los estándares UML y adicionalmente genera complementos adicionales que hacen integral la ayuda al desarrollador no solo durante los ciclos del software si no del proyecto en general.

Fuente: Autores

6.3 MANTENIBILIDAD

Es importante resaltar que la mantenibilidad, es el esfuerzo necesario para diagnosticar deficiencias o fallos, e identificar las posibles modificaciones necesarias en la herramienta. Al momento de identificar fallos en las herramientas que obstaculizan el desarrollo de los diagramas que solución alterna se podrá obtener o como se podrá adaptar fácilmente.

Y si se llegara a provocar alguna modificación que tan sencillo es predecir dichos riesgos, es decir si los fallos se presentan en la fase final como afecta directamente la herramienta y que tan fácil será corregirlos.

Tabla 10. CRITERIOS DE EVALUACION MANTENIBILIDAD

CRITERIOS DE EVALUACION	
VALOR	**DEFINICION**
1	Muy mala
2	Regular
3	Buena
4	Muy buena
5	Excelente

Fuente: Autores

Tabla 11. MANTENIBILIDAD

HERAMIENTAS		CAPACIDAD DE ANALISIS	FACILIDAD DE PRUEBA	ESTABILIDAD	CAPACIDAD DE MODIFICACION
STARUML	CALIFICACION	2	5	5	5
	OBSERVACIONES	• El soporte que ofrece la herramienta cuando ocurren fallos es incompleto. • No brinda un análisis de tiempo de fallos o seguimiento de datos registrados en las operaciones.	Porque al usar versiones antiguas de las herramientas, puesto que al actualizar realizan modificaciones directamente en ella, aun permite hacer pruebas y uso de documentos sin importar la versión en al cual fue desarrollada.	Al permitir modificaciones en su código fuente, permite encontrar un punto de equilibrio ya que se pueden predecir posibles riesgos cuando ocurren efectos que no se prevenían.	Por ser una herramienta libre, da facilidad de modificación en su código fuente, puesto que este es libre también con el fin de que los usuarios aporten a las mejoras del desarrollo y corrección.
POSEIDON FOR UML	CALIFICACION	5	5	5	3
	OBSERVACIONES	Por medio de la edición de la comunidad, facilita el reporte de errores y la rápida atención de Gentleware para solucionar aquellos problemas.	Posee compatibilidad entre versiones por lo tanto realizar pruebas en una versión u otra, permite fácilmente acceder a todas las características de las versiones más actuales sin causar daños en los datos con que se opera.	Posee un gran soporte de estabilidad, modificaciones o actualizaciones se pueden realizar en caliente con un riesgo altamente mitigado	Su licencia no permite modificaciones ya que no es código abierto, sin embargo en la CE (Comunity Edition), permite realizar aportes para futuras versiones de esta herramienta.
ENTERPRISE ARCHITECT	CALIFICACION	5	5	4	2
	OBSERVACIONES	Excelente soporte de la herramienta cuando ocurren fallas, puesto que el soporte está contemplado en su licencia. Brinda análisis y copias de seguridad al momento de fallos, porque este percibe fallos graves y cierra la herramienta pero no se pierde lo hecho, así el usuario no se percate de guardar, la herramienta lo hace.	Es muy fácil hacer pruebas sin importar la versión, aunque la plataforma si influye, porque es una herramienta desarrollada totalmente en Windows. Tiene excelentes complementos de compilación para plataformas diversas.	Soporta los errores en cualquier etapa, sin importar si se modifico la versión. Aunque entre versiones aun surgen errores y dificultades para ciertas modificaciones estructurales, puesto que su código no es abierto.	No posee una capacidad de modificación por parte de los usuarios, solo tiene acceso a ella como tal por parte de los desarrolladores de esta, es una herramienta licenciada, absolutamente completa y con mejoras contantes y progresivas.

Fuente: Autores

6.4 PORTABILIDAD

Como se menciono la portabilidad consiste en la habilidad que tiene el software de funcionar sobre diferentes plataformas, obviamente entendiendo plataforma ya sea arquitectura, sistema operativo, lenguaje de programación o interfaz de usuario.

Para esta evaluación se observa la portabilidad de las tres herramientas, para esto se instala la herramienta en el sistemas operativo Ubuntu 11.10 de 32 bits, para ver que cambios surgen y que tan portable es, al igual que abrir los archivos sobre la herramienta instalada en el sistema operativo Windows 97, XP, 7.

Como se menciono se evalúa cada una de las cuatro características en las que se clasifica la métrica para facilitar la evaluación en la adaptabilidad se tiene en cuenta si al momento de usar la herramienta en otras plataformas sufre modificaciones significativas y para dicha medida se debe instalar en otros ambientes, dicha acción se observa y evalúa detalladamente, debido a que si esta facilidad no se está dando, esta herramienta proporciona baja portabilidad, en cuanto a capacidad de reemplazo, se visualiza y califica que tan bien se adapta a los estándares y que oportunidad ofrece para ser sustituida por productos con funciones similares.

Tabla 12. CRITERIOS DE EVALUACION PORTABILIDAD

CRITERIOS DE EVALUACION	
VALOR	**DEFINICION**
1	Muy deficiente
2	Deficiente
3	Regular
4	Buena
5	Excelente

Fuente: Autores

Tabla 13. EVALUACION PORTABILIDAD

HERAMIENTAS		CAPACIDAD DE REEMPLAZO	FACILIDAD DE INSTALACION	ADAPTABILIDAD
STARUML	CALIFICACION	2	3	5
	OBSERVACIONES	Los archivos .uml que genera Star UML no son legibles por otras herramientas de las usadas en este proyecto, existan dos herramientas compatible, pero que no han sido probadas como parte de este estudio y esto indica un bajo grado de capacidad de reemplazo.	La instalación es un poco más compleja para usuarios inexpertos, porque se debe tener un exhaustivo cuidado con los repositorios del sistema operativo y verificar que este permita la correcta instalación de la herramienta.	La herramienta en Ubuntu no genera modificaciones ni graficas, ni estructurales, conserva sus funcionalidades.
POSEIDON FOR UML	CALIFICACION	2	5	5
	OBSERVACIONES	Los archivos generados en Poseidon For UML son únicamente compatible con versiones diferentes del mismo mas no es posible ejecutarlos o editarlos en otras herramientas similares. A excepción del software en el cual está basado Poseidon For UML.	Simple de instalar, Gentleware ofrece versiones para Windows, MacOS X y Linux, para S.O. de 32 y 64 bits. Para la instalación no es necesario tener una licencia, más si para guardar algún proyecto realizado.	Puesto que es desarrollado en Java su portabilidad es total, ya que la maquina virtual de Java permite que el sistema se ejecute bajo cualquier otra plataforma.
ENTERPRISE ARCHITECT	CALIFICACION	4	3	5
	OBSERVACIONES	Los archivos .EA solo son compatibles con la herramienta, pero esta permite generar archivos xml, html y XMI.	Para Ubunto posee una versión de prueba que se instala fácilmente pero debe incluir complementos para que funcione igualmente, puesto que esta es una herramienta desarrollada en Windows.	A pesar de ser desarrollado en Windows posee una alta portabilidad, tiene uso multiusuario.

Fuente: Autores

6.5 USABILIDAD

Esta medida es altamente influyente para tomar una decisión si la herramienta es acorde o no dependiendo las necesidades del usuario, esta es considerada una de las más importantes y relevantes en este proyecto, puesto que las herramientas CASE son diseñadas especialmente para analistas y desarrolladores de software con el fin de lograr un objetivo especifico.

Cuando se habla de usabilidad concretamente se está haciendo referencia a la relación usuario-herramienta, como se menciona anteriormente basados en la ISO 9126 se evalúa el esfuerzo necesario para conocer la herramienta y la estructura para aprender a usarla y su Operabilidad que para el uso de este proyecto se medirá directamente el control del sistema. Con respaldo en la ISO 9241-11 la cual sirve de guía para identificar los aspectos más importantes para evaluar la usabilidad pero orientados en términos de desempeño y satisfacción del usuario.

Por definición la comprensión se describe como un proceso de creación mental, por el cual partiendo del emisor en este caso la herramienta, los usuarios quienes son los receptores crean una imagen de lo que realmente quieren transmitir, es decir lo que realmente se mide es que tan buena es la herramienta para lograr la mejor imagen en los usuarios, es decir que tan fácil se puede interpretar la herramienta[15].

Tabla 14. CRITERIOS DE EVALUACION USABILIDAD

CRITERIOS DE EVALUACION	
VALOR	**DEFINICION**
1	Muy deficiente
2	Deficiente
3	Regular
4	Buena
5	Excelente

Fuente: Autores

[15]Definición de comprensión
http://www.santurtzieus.com/gela_irekia/materialak/ikastaro/comprender/ulermena/ulertzea.htm

Tabla 15.USABILIDAD

HERAMIENTAS		COMPRENSIÓN	OPERABILIDAD	FACILIDAD DE APRENDER
STARUML	CALIFICACION	4	5	4
	OBSERVACIONES	El esfuerzo para conocer la herramienta y su operación es relativo, dependiendo del usuario, pero en general un usuario común fácilmente se adaptara a la interfaz de sus herramientas y componentes.	La operación del sistema es fácil, tiene posibilidades de adaptación a redes de área local, puesto que es una herramienta libre y facilita su uso y control.	El esfuerzo no es tan alto, puesto que si el usuario posee conocimientos básicos en UML, la herramienta cuenta con los componentes similares y precisos para esta, en general su uso es fácil de percibir y comprender mas no organizado.
POSEIDON FOR UML FOR UML	CALIFICACION	5	5	5
	OBSERVACIONES	Es un software intuitivo por excelencia, presenta al usuario una GUI muy sutil con herramientas que le dicen al usuario cual es el próximo paso a seguir, aunque al igual que todo este tipo de herramientas, es necesario conocer mínimamente los conceptos básicos de UML para sacarle provecho.	Es de Uso Muy simple tanto que si los diagramas base están bien fundamentados, a partir de ellos sale casi el 80% de los diagramas que de él se derivan, pidiéndole al usuario únicamente arrastrar los diferentes modelos o conectores, y simplemente seleccionando las identidades de cada uno de los objetos.	Dicho antes esta herramienta es intuitiva por lo que permite al usuario realizar las tareas más simples, casi en su primer contacto con la herramienta.
ENTERPRISE ARCHITECT	CALIFICACION	2	4	3
	OBSERVACIONES	La comprensión lógica e la estructura es compleja, a pesar de contar con ayudas visuales de cada componente, no es tan claro asumir lo pasos correctos para aprovechar al máximo la aplicación. El usuario debe aplicar un esfuerzo más disciplinado y constante para aprender la estructura lógica y funcional de Enterprise Architect.	Controlar la herramienta en sus sistema como tal puede llegar a ser un poco más complejo ya que por la licencia dificulta el hospedaje y operación de esta.	La herramienta es muy completa, pero un poco más compleja, para usuarios con poco conocimiento en objetos y UML, puesto que cuenta con un completo menú de herramientas para todo tipo de desarrollo, en especial para UML, cada componente del diagrama cuenta con propiedades y dependencias que pueden no ser tan claras, cuando no se manejan los conocimientos y términos dentro del nivel de diseño de software. Para aprender a usar correctamente la herramienta se necesita un grado alto de esfuerzo y compromiso, porque un usuario la puede usar a su modo, es decir adaptándola a sus facilidades, pero estará desaprovechando los beneficios que brinda la herramienta y dificultara hacer mejoras, o la generación de código no será tan acertada.

Fuente: Autores

6.6 EFICIENCIA

De la definición de la real academia de la lengua la cual define la eficiencia como la capacidad de disponer de alguien o de algo para obtener un efecto deseado[16]; el cual se aplica a este proyecto, porque partiendo de esta definición se aplica la evaluación puesto que en cuestiones de la herramienta se mide en referencia a que tan eficiente es la herramienta con respecto a dos aspectos específicos, el tiempo y los recursos.

Con respecto al tiempo especialmente se observa el tiempo que tarda la herramienta en generar el código con respecto a los diagramas, el tiempo de procesamiento de cada herramienta, y que tanto demora la construcción de los diagramas, en general es medir los tiempos de la herramienta.

Y con respecto a los recursos este proyecto se limita a prestar atención que tanto cumple con los estándares de la UML para el diseño de los diagramas, es decir que tenga los recursos necesarios para su construcción. En cuestión de los recursos de hardware solo se tiene en cuenta la capacidad que consumen en memoria.

Tabla 16. CRITERIOS DE EVALUACION EFICIENCIA

CRITERIOS DE EVALUACION	
VALOR	**DEFINICION**
1	Muy deficiente
2	Deficiente
3	Regular
4	Buena
5	Excelente

Fuente: Autores

[16] Diccionario real academia de la lengua www.rae.es

Tabla 17. TABLA DE EVALUACION EFICIENCIA

EFICIENCIA			
HERAMIENTAS		**RESPECTO AL TIEMPO**	**RESPECTO A LOS RECURSOS**
STARUML	**CALIFICACION**	**4**	**4**
	OBSERVACIONES	En cuanto a tiempos de uso, la herramienta es regular ya que genera bloqueos fácilmente al momento de generar código, para generar imágenes y diagramas la herramienta es rápida, su tiempo de respuesta está entre 1 a 3 segundos aproximadamente.	Como se menciona anteriormente en las especificaciones en cuanto al espacio que ocupa en el disco duro es de 110 mb y en cuanto a los recursos propios es decir, sus herramientas cumplen con lo establecidos por los estándares UML, con lo dicho se considera que posee relativamente una buena eficiencia pero que podría mejorar en gran medida.
POSEIDON FOR UML	**CALIFICACION**	**5**	**5**
	OBSERVACIONES	Carga y se ejecuta en alrededor de 3 segundos, exporta diagramas a componentes gráficos como Jpg, Png y BMP en alrededor de 0.5 a 1 segundo, además de que la generación de diagrama con elementos heredados es casi inmediata.	Se ejecuta en aproximadamente 200KB de memoria física, además se ejecuta directamente en la maquina virtual de Java por lo que garantiza de forma eficiente protección de memoria en procesadores que no poseen gestor de memoria (MMU).
ENTERPRISE ARCHITECT	**CALIFICACION**	**4**	**5**
	OBSERVACIONES	• La herramienta es muy rápida, pero constantemente se bloquea, lo cual retarda los tiempos de respuesta, entre un diagrama y otro, al generar diagramas a partir de cada caso de uso, su tiempo de respuesta en milisegundos pero al guardar estos o exportarlos como imagen puede llegar a tardar un poco mas y en ocasiones cerrar la herramienta debido a fallas y esto aumenta la eficiencia pero negativamente. • Los tiempo de respuestas en modelos grandes son muy positivos gracias a su ventana de trazabilidad.	• La más completa con respecto a los recursos que proporciona, sus recursos son completos en cuanto a idiomas y lenguajes que soporta. • Además se adapta a los últimos estándares de StarUML 2.1 para modelado avanzado. Esta herramienta proporciona la trazabilidad a los proyectos robustos de software, por su completitud durante todos los ciclos de vida del software es recomendada para grandes desarrollos ya que se puede dar un uso al 100% de sus recursos.

Fuente: Autores

6.7 INTEGRIDAD

Definido anteriormente la integridad de software es la capacidad de un sistema o herramienta de mantener sus datos intactos ante la aparición de fallos o intrusiones invasivas. Es así como se evalúa la integridad en estas herramientas tanto en como mitigan las perdidas ante la presencia de fallos, y protegen los datos de eliminaciones involuntarias y/o ajenas al usuario.

TABLA 18. CRITERIOS DE EVALUACION INTEGRIDAD

CRITERIOS DE EVALUACION	
VALOR	**DEFINICION**
1	Muy deficiente
2	Deficiente
3	Regular
4	Buena
5	Excelente

Fuente: Autores

Tabla 19. EVALUACION INTEGRIDAD

INTEGRIDAD		
HERAMIENTAS		
STARUML	**CALIFICACION**	**3**
	OBSERVACIONES	En ocasiones falla la plataforma Win32, en ambientes de 64bits sin razón aparente, que son los más comunes en esta época, a raíz de ello se obtienen pérdidas de los datos que se estén operando y que no hayan sido almacenados.
POSEIDON FOR UML	**CALIFICACION**	**5**
	OBSERVACIONES	Mantiene las diferentes componentes del modelo integradas entre sí, permitiendo que los cambios en ellas se generen o hereden automáticamente en los diagramas afectados. garantizando así una integridad referencial entre los diferente diagramas
ENTERPRISE ARCHITECT	**CALIFICACION**	**5**
	OBSERVACIONES	Por su fuerte estructura basada en las especificaciones UML 2, lo cual facilita la integridad de los diagramas cuando unos heredan de otros, en general toda la estructura en este proceso de evaluación de Enterprise fue heredada de los casos de uso, y puesto que EA utiliza perfiles UML para extender el dominio asegura en su totalidad la integridad de los datos.

Fuente: Autores

Tabla 20. CALIFICACION DE LAS HERRAMIENTAS EN PROMEDIOS Y PORCENTUALES

HERRAMIENTA / METRICA	STARUML		POSEIDON FOR UML		ENTERPRISE ARCHITECT		
	Promedio						TOTAL
	Total	%	Total	%	Total	%	
Fiabilidad Medición dependiente de la cantidad de fallos de una herramienta de software en un tiempo determinado.	4	80	5	100	5	100	**4,67**
Funcionalidad Nivel en el que una herramienta de software se desempeña como sus desarrolladores prometen	4,4	88	4,4	80	5	100	**4,6**
Mantenibilidad Grado en el que la herramienta de software permite reportar y/o modificar fallos e inconvenientes.	4,3	85	4,5	90	4	80	**4,27**
Portabilidad Capacidad de ejecutarse y mantener sus funciones en diferentes plataformas.	3,3	67	4	80	4	80	**3,77**
Usabilidad Nivel de satisfacción que le ofrece la herramienta, combinado con la facilidad de adaptación, desde el punto de vista del usuario	4,33	86,66	5	100	3	60	**4,11**
Eficiencia Comportamiento con respecto al tiempo y a los recursos.	4	80	5	100	4,5	90	**4,5**
Integridad Capacidad de Conservar y proteger los datos frente a la aparición de fallos o intrusiones no procedentes del usuario.	3	60	5	100	5	100	**4,67**
NIVEL TOTAL DE SATISFACCION DE CADA HERRAMIENTA	**4,05**	**78,09**	**4,70**	**92,86**	**4,36**	**87,14**	**4,37**

Fuente: Autores

ANALISIS Y CONCLUSIONES DE LA TABLA DE CALIFICACION DE LAS HERRAMIENTAS EN PROMEDIOS Y PORCENTAJES

Analizando la anterior tabla de manera porcentual se puede establecer:

- Respecto al nivel de fiabilidad evaluado, Poseidón y Enterprise Architect son más acertadas que StarUML, esto a partir de que estas dos cuentan con licencias y soporte de fallos lo cual genera la corrección a tiempo, y StarUML es un poco más baja en su calificación, aunque en la web se encuentran muchos foros y páginas de usuarios que aportan corrección al código es absolutamente libre y esto provoca un margen de error mayor.

- Enterprise Architect es la que proporciona la mejor funcionalidad, seguida de StarUML y por ultimo esta Poseidon For UML, EA es la herramienta que cumple con lo que los proveedores ofrecen, muy completa para grandes desarrollos y en especial para proyectos integrales, puesto que aporta gran material en desarrollo de proyectos de negocio.

- Poseidon For UML a pesar de no ser una herramienta absolutamente libre, ofrece mejor mantenibilidad que las otras, seguida por StarUML la cual es favorecida por su libertad en la modificación de código, para StarUML aunque su código libre es su debilidad en la fiabilidad, aporta gran fortaleza en mantenibilidad.

- En portabilidad aun se encuentran algunas falencias, pero se tienen dos opciones muy acertadas y casi en su totalidad altamente portables que son Poseidon For UML y Enterprise Architect, debido a que ambas cuentan con la capacidad de ejecutarse tanto en la plataforma Ubuntu 11.10 como en Windows 7, Xp; que fueron las plataformas utilizadas en este proyecto, mantiene sus funciones en un bajo grado de cambio, el mayor obstáculo de Poseidon For UML se observa en el grado de capacidad de reemplazo, el cual obtiene una calificación de 2 con respecto a Enterprise que es de 4, pero esto es compensado en la facilidad de instalación puesto que Poseidon For UML es muy fácil de instalar y EA presenta un grado más alto de dificultad en este aspecto, este tipo de balances son los que proporcionan que las dos herramientas en su promedio porcentual obtenga el mismo valor de 80%. La portabilidad en StarUML es más baja, puesto que este cuenta con un factor diferenciador que le genera un valor en contra, el cual es que para su instalación debe usar WINE, y este es una maquina virtual para Windows, es decir será una maquina virtual de Windows ejecutándose sobre Linux, por sus parte Poseidon For UML si cuenta con un instalador diferenciando la plataforma: Linux o Windows.

- La herramienta con mayor índice de usabilidad es Poseidon For UML, seguida de StarUML y por ultimo Enterprise Architect por ser muy completa pero de más difícil aceptación y uso, las dos herramientas con mejor porcentaje que son Poseidon For UML y StarUML se debe a que sus entornos gráficos son más cercanos a los parámetros del estándar de UML, y esto puede resultar más familiar al usuario, aunque en Enterprise Architect opera los estándares UML, exige un grado mayor de conocimiento en diseño orientado a objetos.

- La mayor eficiencia se presenta en Poseidon For UML con un 100% ya que su comportamiento con respecto al tiempo es el menor en tiempos de respuesta y el consumo de recursos es mínimo con respecto a las otras dos herramientas,

gracias a que ocupa menos memoria física mientras se ejecuta en la máquina virtual de JAVA, lo cual no ocurre en las otras dos herramientas.

- Y al igual que en fiabilidad de nuevo se encuentra una igualdad en cuanto a integridad, entre Poseidon For UML y Enterprise, dando una probabilidad del 100% de confianza en estas herramientas las cuales desde el punto de vista particular, se considera que tienen integridad total. Ambas poseen la capacidad de protección y conservación de los datos en el momento que llegase a ocurrir algún fallo no controlado o instrucciones físicas o virtuales ajenas al usuario.

CONCLUSIONES

En el cambiante universo del desarrollo de software, es necesario utilizar herramientas que se acomoden a las necesidades al momento de seleccionarlas y a su vez se ajusten a medida que se desarrolla un proyecto, para que el usuario, en este caso el desarrollador, tenga a disposición el acompañamiento de entidades, tales como comunidades o empresas que le permitan solucionar a tiempo problemas ocasionados en las fases mas criticas del desarrollo de software. Es así como es posible observar que las herramientas libres son una opción muy favorable, tanto en aspectos económicos o presupuestales, como en ventajas de uso al momento de elegir entre los diferentes tipos de software que ofrece el mercado, aunque a veces la búsqueda de ayuda u orientación se vuelve exhaustiva ya que entre las distintas comunidades de software libre se puede encontrar miles de millones posibles soluciones o ayudas, pero esto en ocasiones quita tiempo, y en el desarrollo de software , como en la mayoría de los campos de negocio, el tiempo es dinero. También existen herramientas que por un pago reducido ofrecen miles de ventajas y ahorran tiempo al momento de solucionar problemas. Es así el caso de Poseidon For UML, que por la rentar una licencia un solo mes, tienes acceso a una gran comunidad dispuesta a colaborarte en tus necesidades, sin embargo El caso de STAR UML, muestra que miles de millones de comunidades, pueden aportar un gran valor agregado a tus proyectos, además de permitir compartir tus experiencias y necesidades con toda la web. Y el caso de Enterprise Architect que aunque su costo es más elevado, es totalmente completa en cuanto a herramientas y soporte.

Este estudio no desmerita ninguna de las tres herramientas por el contrarió destaca las fortalezas de cada una, concluyendo que si el proyecto es muy robusto y los usuarios poseen conocimientos avanzados en desarrollo de proyectos, diseño de software la herramienta más acertada para este caso es Enterprise Architect, entre más grande y complejo el proyecto mayor aprovechamiento de esta herramientas. Si el proyecto es más sencillo o por lo menos el usuario es novato en el uso de herramientas UML Poseidon For UML es acorde, puesto que la más agradable y clara para los usuarios y contiene capacidades de autogeneración, es decir con pocos datos proporciona excelentes modelos. StarUML es una buena herramienta con un nivel bueno en la evaluación como promedio total de las métricas evaluadas, por ser libre permite mejoras futuras pero aun falta trabajo en sus herramientas y visualmente puede ser desordenada, es acorde para usuarios principiantas y proyectos pequeños y educativos.

BIBLIOGRAFIA

[1] Beatriz Marín. Giovanni Giachetti. Oscar Pastor. Intercambio de modelos UML y OO-Method1. Departamento de Sistemas Informáticos y Computación Universidad Politécnica. Valencia, España <http://kuainasi.ciens.ucv.ve/ideas07/documentos/articulos_ideas/Articulo43.pdf> [En Línea] [Citado 23 mayo de 2011]

[2] Carlos M. Zapata J. Fernando Arango I. Raquel Anaya de Páez. Estudio comparativo de las herramientas MetaCASE bajo consistencia y refinamiento. Medellín, Colombia Enero de 2007

[3] Guía técnica sobre evaluación de software en la administración pública, Perú http://www.ongei.gob.pe/bancos/banco_normas/archivos/Guia-Evaluacion-SW.pdf[En Línea] [Citado 25 Noviembre 2011]

[4] Criterios para seleccionar y evaluar un software de mantenimiento (Revista Mantenimiento - Chile - Nº 26 - AÑO 1996 – ISS 0716 – 8616) http://www.mantenimientomundial.com/sites/mm/notas/criterios.pdf[En Línea] [Citado Noviembre de 2011]

[5] Metodología de diseño, desarrollo y evaluación de software educativo Tesis de Magíster en Informática. Facultad de Informática. UNLP Ing. Zulma Cataldi Directores: Dr. Ramón García-Martínez // Ing. Raúl Pessacq //ISBN 960-34-0204-2 // 2000 http://laboratorios.fi.uba.ar/lsi/cataldi-tesisdemagistereninformatica.pdf [En Línea] [Citado Octubre de 2011]

[6] TECNICASDE EVALUACION DE SW Versión: 12.0 Fecha: 17 de octubre de 2005 Autoras: Natalia Juristo, Ana M. Moreno, Sira Vegas http://is.ls.fi.upm.es/udis/docencia/erdsi/Documentacion-Evaluacion-6.pdf[En Línea] [Noviembre de 2011]

[7] Usability NET. International Estándar. Usabilidad http://www.usabilitynet.org/tools/r_international.htm#9241-1x

[8] Balderrama. Jose O. Información Tecnológica Vol. 10. Chile 1999. Pag 327. http://books.google.com.co/books?id=Z0fUgdnVHdgC&pg=PA327&dq=que+son+herramientas+CASE&hl=es&sa=X&ei=lBweT8jzONCFsAKdhoHMDg&ved=0CDgQ6AEwAg#v=onepage&q=que%20son%20herramientas%20CASE&f=false

[9] Página oficial StarUMLhttp://staruml.sourceforge.net/en/

[10] Pagina de Enterprise Architect distribuidores http://www.sparxsystems.com/products/ea/index.html

[11] Diccionario real academia de la lengua www.rae.es

[12] Definición de comprensión http://www.santurtzieus.com/gela_irekia/materialak/ikastaro/comprender/ulermena/ulertzea.htm

[13] Definición de Eficiencia http://informacion.wordpress.com/2006/06/06/%C2%BFque-son-los-recursos-informaticos/

[14] Abud Figueroa. María Antonia. Calidad en la Industria del Software. La norma ISO-9126. Revista Upiicsa. Enero a Abril 2004. http://www.revistaupiicsa.20m.com/Emilia/RevEneAbr04/Antonieta1.pdf[En Línea] [Citado Octubre de 2011]

[15] Daniel Cabarcas. Fernando Arango. Carlos M. Zapata. Establecimiento y verificación de la consistencia en dome: un caso de estudio. Dynarev.fac.nac.minas vol.73 no.148 Medellín Enero. 2006. Escuela de Sistemas. Facultad de Minas. Universidad Nacional de Colombia, sede Medellín. <http://www.scielo.unal.edu.co/scielo.php?pid=S0012-73532006000100003&script=sci_arttext&tlng=es> [En Línea] [Citado 23 mayo de 2011]

[16] Gonzalo Génova Fuster, José Miguel Fuentes Torres, María Cruz Valiente Vázquez. Evaluación comparativa de herramientas CASE para UML desde el punto de vista notacional. Novática: Revista de la Asociación de Técnicos de Informática, ISSN 0211-2124, Nº. 181, 2006 (Ejemplar dedicado a: Las licencias de Software Libre y su contexto). p 59-64. <http://dialnet.unirioja.es/servlet/articulo?codigo=2065795> [En Línea] [Citado 23 mayo de 2011]

[17] Ing. Jesús Alberto Ochoa. Herramientas CASE. Universidad Cooperativa De Colombia Espinal – Tolima <http://members.fortunecity.com/software1/herramie.htm>. [En Línea] [Citado 23 mayo de 2011]

[18] Juan Bernardo Quintero. Raquel Anaya de Páez. Marco de Referencia para la Evaluación de Herramientas Basadas en MDA. Grupo de Investigación en Ingeniería de Software, Universidad EAFIT. Medellín, Colombia <http://kuainasi.ciens.ucv.ve/ideas07/documentos/articulos_ideas/Articulo65.pdf> [En Línea] [Citado 23 mayo de 2011]

[19] Mauro Callejas Cuervo. Oscar Yovany Baquero Moreno. Herramientas libres para modelar software. Revista Facultad de Ingeniería CEDEC 14(19), 2 de Diciembre de 2005

[20] Teresita Rojas. Luis E. Mendoza. María A. Pérez. Modelo De Decisión Para Soportar La Selección De Herramientas CASE Venezuela <http://www.lisi.usb.ve/publicaciones/05%20herramientas/herramientas_09.pdf> [En Línea] [Citado 23 mayo de 2011]

[21] Gonzales Doria. Heidi. Las métricas de software y su uso en la región. Cholula, Puebla. México. Mayo 7 de 2001. Capitulo 3. http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/gonzalez_d_h/ [En Línea] [Citado Marzo de 2012]

[22] Moya Cruz, Sandra Orfelina (2010). Estudio comparativo de las herramientas raitonal rewu site procaliberrm, y doors utilizadas durante la etapa de análisis de requerimientos. Facultad de Ingeniería en Sistemas e Informática. ESPE. Sede Sangolquí <http://www3.espe.edu.ec:8700/handle/21000/311> [En Línea] [Citado 23 mayo de 2011]

LIBROS

[23] Cantone, Dante. LA BIBLIA DEL PROGRAMADOR. IMPLEMENTACION Y DEBUGGING. Buenos Aires Argentina (2007)

[24] F. Alonso Amo. Loïc Martínez Normand. Introducción a la ingeniería del software. España (2005) p 78-91.

[25] Kenneth E. Kendall, Julie E. Kendall, Antonio Núñez Ramos. Análisis y diseño de sistemas. Sexta Edición. México 2005. p.14 - 15

[26] Peter Rob. Carlos Coronel. Sistemas de bases de datos: diseño, implementación y administración. México. Enero (2006) p 780