UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad #9

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN INFORMÁTICA

TÍTULO: Subsistema de componentes, modelos y propiedades de la plataforma para el desarrollo de simuladores de procesos químicos, del proyecto de Simulación de la Facultad 9.

AUTOR(ES): Mayté Solano Milanés
Adrian López Rosales

TUTOR: Ing. Yurisnel Corrales Valdés

Ciudad de La Habana, junio 30 de 2010.
Año 52 de la Revolución.

"El conocimiento se adquiere por medio del estudio; la sabiduría, por medio de la observación. "

Marilyn vos Savant

DECLARACIÓN DE AUTORÍA

Por este medio se declara que somos los únicos autores de este trabajo y se autoriza a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Adrian López Rosales

_____

(Firma del autor)

Mayté Solano Milanés

_____

(Firma del autor)

Ing. Yurisnel Corrales Valdés

_____

(Firma del tutor)

En la industria química del siglo XXI para poder insertarse en el mercado mundial es preciso perfeccionar los procesos industriales de manera continua. La simulación de procesos es considerada un instrumento fundamental en la optimización del trabajo en diversas ramas de la ingeniería química.
En Cuba se le ha aplicado la simulación a varios procesos químicos empleando un conjunto de técnicas utilizadas internacionalmente. El proyecto de Simulación de la facultad #9 en la Universidad de las Ciencias Informáticas necesita de una herramienta que permita gestionar los componentes químicos, propiedades físico-químicas y modelos de operaciones unitarias, tarea indispensable en la simulación.
En el presente documento se plasman los resultados del estudio realizado para el desarrollo de dicha solución, se explican los conceptos relacionados con el mismo, se realiza el modelado e implementación de la solución propuesta, finalmente se dejan algunas recomendaciones para el mejoramiento futuro de la propuesta desarrollada.

Palabras claves: simulación de procesos, componentes químicos, propiedades físico químicas, operaciones unitarias.

INTRODUCCIÓN

La Simulación de Procesos juega un papel muy importante en la industria química como herramienta para el diseño, caracterización, optimización y monitoreo del funcionamiento de procesos industriales. En el mercado actual existe una gran competitividad, por lo que se hace indispensable el constante perfeccionamiento de los procesos industriales, principalmente los que influyen en los costos de operación. En este sentido la simulación posee un gran número beneficios para diversas ramas de la ingeniería química encaminados a hacer más rápido y con mayor calidad el trabajo, algunas de ellas son:

- Detección de cuellos de botellas en la producción.
- Predicción de los efectos de cambios en las condiciones de operación y capacidad de la planta.
- Optimización de las variables de operación.
- Optimización del proceso cuando cambian las características de los insumos y/o las condiciones económicas del mercado.
- Análisis de nuevos procesos para nuevos productos.
- Evaluación de alternativas de procesos para reducir el consumo de energía.
- Análisis de condiciones críticas de operación.
- Transformación de un producto para desarrollar otras materias primas.
- Análisis de factibilidad y viabilidad de nuevos procesos.
- Optimización del procesos para minimizar la producción de desechos y contaminantes.
- Entrenamiento de operadores e ingenieros de proceso.
- Investigación de la factibilidad de automatización de un proceso. **(1)**

Existe una gran variedad de simuladores de procesos, anteriormente solo lo usaban los ingenieros que diseñaban procesos, ahora ingenieros con poco o ninguna instrucción de programación puede modelar procesos complejos.

Los primeros pasos de la simulación de procesos en la Ingeniería Química se basa principalmente en circuitos analógicos, que demuestran que diversos principios físicos tienen asociados modelos matemáticos equivalentes. Posteriormente a partir del uso masivo de la computadora digital se evoluciona lentamente de la simulación analógica a la digital, desapareciendo prácticamente esta primera.
Con el empleo de los ordenadores en las universidades en los años sesenta comenzó la simulación por ordenador y a partir de ahí se generaron una sucesión de acontecimientos que permitieron que en la actualidad existan eficientes simuladores comerciales como por ejemplo HYSYS, ASPEN PLUS, CHEMCAD, SUPERPRO DESIGNER y otros.

Estos simuladores de renombre son sobretodos productos de transnacionales vinculados a esta industria y universidades de primer nivel mundial, debido a que construir un modelo de simulación es costoso, principalmente porque debe ser verificado y validado.
Organizaciones como COLAN (CAPE-OPEN Laboratories Network) promueven el uso de estándares abiertos para software de simulación de procesos con el objetivo de permitir la interoperabilidad de simuladores de procesos y de modelos especializados para operaciones unitarias, paquetes termodinámicos y paquetes numéricos.

El Standard CAPE-OPEN (Computer-Aided Process Engineering), es una plataforma que ofrece integración de componentes para la simulación de procesos físicos y químicos. Es multiplataforma y utiliza importantes marcos de trabajo y software de conectividad como CORBA, COM, J2EE, EJB, etc., orientada a componentes, tiene una concepción de interfaces orientadas a objetos y usa técnicas de reingeniería y encapsulación. Entre los ambientes para la simulación de procesos desarrollados sobre esta plataforma están ProsimPlus de la compañía ProSim SA, simulador estacionario para la producción de oxígeno, INDISS de RSI, este último para la simulación dinámica y otros como Aspen Engineering Suite, Process Engineering Suite, HYSYS, GPROMS. (2)

Esta plataforma (CAPE-OPEN) es financiada por la Unión Europea y el Programa británico Brite-EuRam III y en 1997 eran parte del Proyecto más de 50 personas de empresas transnacionales e instituciones académicas. Tomando como ejemplo dentro de los integrantes solo las universidades, se puede resaltar que ocupan posiciones en un ranking internacional entre las mejores 500 universidades del mundo.

En Cuba han sido aplicados simulaciones a los procesos de las producciones de azúcar crudo y refino, alcohol, ácido sulfúrico, sosa cáustica, refinación de petróleo, níquel, cobalto y otras. El empleo de programas de simulación en la industria química cubana comenzó en el ISPJAE en el año 73 bajo la tutoría de Crowe.C.M autor del libro “Chemical Plant Simulation” y de los creadores del programa

denominado “General Engineering and Management Computation System (GEMCS)” desarrollado en la Universidad de McMaster, Canadá.

En el período de 1973 – 1974 el GEMCS se logró aplicar al proceso azucarero en las computadoras disponibles en Cuba, gracias a estudiantes del IPSJAE con cuatro tesis de maestría.

Luego surgió TERMOAZUCAR V.1 en la Universidad de Cienfuegos con la cooperación de IPROYAZ (Instituto de Proyectos Azucareros). Las últimas versiones del TERMOAZUCAR en MS-DOS son la V 3.1 que contienen diversas mejoras en contenido por la adición de modelos que no aparecen en versiones anteriores. El TERMOAZUCAR es el único simulador cubano o extranjero que se puede enlazar coherentemente con un Programa Experto computarizado (ANSTE) asegurando la calidad de las soluciones que se alcanzan.

La estructura y programación flexible empleada en el TERMOAZÚCAR permitió sin grandes dificultades transferir lo disponible a una nueva versión a partir del trabajo coordinado del ISPJAE y la UCI. Así se creó el Sistema de Termo Azúcar (STA) que supera las limitaciones anteriores, como la vista de programación y la posibilidad del empleo coherente de un conjunto de técnicas empleadas internacionalmente.

Pero todo lo logrado hasta ahora en el tema de simuladores en el país fue imponiéndose ante fuertes limitaciones que frenan el desarrollo de la industria del software cubano. El bloqueo estadounidense es una de las trabas que inciden directamente en el uso social e intensivo de los recursos TIC trayendo consigo restricciones económicas, tecnológicas y de comunicación. Los software extranjeros solo pueden utilizarse para fines académicos por el alto gasto financiero que requieren, convirtiéndose la independencia tecnológica nacional en la solución para producir competitivamente y mejorar la posición en el mercado mundial.

Surge entonces como **situación problemática** que el proyecto de Simulación de la facultad #9 en la Universidad de las Ciencias Informáticas tiene como objetivo la creación de una plataforma para el desarrollo de simuladores de procesos químicos para diversos entornos industriales. Dicho proyecto se encuentra inmerso en la primera etapa de desarrollo y no cuenta con una herramienta que permita realizar las operaciones matemáticas fundamentales con los componentes químicos, propiedades físico-químicas y modelos de operaciones unitarias para poder llevar a cabo la simulación. Ante la situación anterior surge como **problema científico** necesidad de incorporar las tareas básicas a la plataforma del proyecto Simulación en la facultad 9 de la Universidad de Ciencias Informáticas, que permitan operar con los componentes químicos, propiedades físico-químicas y modelos de operaciones unitarias.

Para dar solución al problema planteado se traza como **objeto de estudio** los simuladores de procesos químicos. Este determina el **campo de acción** subsistemas de componentes químicos, propiedades

físico-químicas y modelos de operaciones unitarias y define el **objetivo general** desarrollar el subsistema de la plataforma del proyecto Simulación en la facultad 9 de la Universidad de Ciencias Informáticas que permita la gestión de los componentes químicos, propiedades físico-químicas y modelos de operaciones unitarias.

Para llevar a cabo la investigación se establecen las siguientes tareas:

- Caracterizar principales simuladores de procesos químicos existentes.
- Seleccionar la metodología y herramientas a utilizar en el desarrollo del subsistema en cuestión.
- Especificar los requerimientos funcionales y elaborar el modelo del dominio, análisis, diseño e implementación del subsistema, así como aplicar método de prueba de caja blanca.

Con la finalidad de orientar la investigación se parte de la **idea a defender**: El desarrollo de un subsistema de componentes químicos, propiedades físico-químicas y modelos de operaciones unitarias proveerá al proyecto de Simulación de la facultad #9 de la herramienta fundamental para la gestión de los componentes químicos, propiedades físico-químicas y modelos de operaciones unitarias.

En la realización de la investigación se utilizaron los siguientes métodos científicos:

Dentro de los métodos teóricos se empleo el analítico - sintético para buscar los rasgos que caracterizan y distinguen los simuladores de procesos químicos y el histórico - lógico para constatar teóricamente la evolución de su desarrollo.

Como métodos empíricos se realizarán entrevistas y observaciones para la recopilación de los requerimientos que tendrá el subsistema utilizando diferentes puntos de vistas.

Este trabajo de diploma está conformado por cuatro capítulos, que están estructurados de la forma siguiente:

- En el Capítulo1 “Fundamentación teórica”, se realiza una caracterización de los principales simuladores de procesos existentes, sus conceptos, ventajas, desventajas y aspectos generales. Se presenta además un estudio de las metodologías de desarrollo y de las posibles herramientas a utilizar eligiendo las que se adecuen a las características del producto.
- En el Capítulo 2 “Presentación de la solución propuesta” se realiza el modelo del dominio, la especificación de los requerimientos funcionales y no funcionales del subsistema y se presenta el diagrama de casos de usos de sistema.
- En el Capítulo 3 “Construcción de la solución propuesta” se realiza el modelo de diseño, implementación y se realizan las pruebas.

CAPÍTULO 1: Fundamentación Teórica.

Introducción

En este capítulo se abordan conceptos fundamentales como son simulación de proceso y modelo matemático. Otro aspecto a tratar es la caracterización de importantes simuladores de procesos químicos tanto nacional como internacional. Dentro de los simuladores extranjeros se analizaron el HYSYS, SuperPro Designer y CHEMCAD. El Sistema de Termo Azúcar (STA) por ser uno de los últimos simuladores de procesos creados en el país se eligió para un estudio profundo como apoyo a esta investigación. Además se realiza la elección de la metodología de desarrollo y de las herramientas a utilizar.

Simulación

La simulación por computadora constituye una potente técnica de análisis de procesos bien establecida en la Ingeniería Química, con gran actualidad y perspectivas, y con múltiples aplicaciones en el campo de ahorro de energía y en la disminución de la agresión al medio ambiente.

El concepto de simulación hizo su aparición a principio de los años 1950 cuando se le dio importancia al proceso de dividir en partes un problema para examinar la interacción simultánea de todas ellas.

¿Qué es simular?

Simular es condicionar los valores de un modelo determinado para que logre un comportamiento real. Validación de un modelo.

En otras palabras proponer ciertos valores de entrada al programa simulador para obtener resultados o valores de salida, que estiman el comportamiento del sistema real bajo esas condiciones.

La simulación es un método para acercarse a la realidad con gran utilidad para propósitos educacionales, de capacitación y de investigación. **(3)**

Según Shannon es el proceso de diseñar un modelo de un sistema real y llevar a término, experiencias con él, con la finalidad de comprender el comportamiento del sistema o evaluar nuevas estrategias —dentro de los límites impuestos por un cierto criterio o un conjunto de ellos—para el funcionamiento del sistema.**(4)**

Esencialmente la simulación es una técnica numérica que imita el funcionamiento de un proceso del mundo real en un tiempo determinado, con el propósito de explicar, entender o mejorar el proceso.

Tipos de herramientas de simulación

- Simulación cualitativa: Tiene por objeto principalmente el estudio de las relaciones causales y las tendencias temporales cualitativas de un sistema, como así también la propagación de perturbaciones a través de un proceso dado.
- Simulación cuantitativa: Es aquella que describe numéricamente el comportamiento de un proceso, a través de un modelo matemático del mismo.
    - ✓ Simulación estacionaria: Implica resolver los balances de un sistema, no involucrando la variable temporal, por lo que el sistema deseará reflejar en el modelo, las variaciones de interés con las coordenadas espaciales (modelos a parámetros distribuidos) y deberá utilizarse un sistema de ecuaciones según el número de coordenadas espaciales consideradas. **(5)**

De la clasificación anterior se puede aclarar que un estado estacionario es aquel donde los valores de las variables no cambian con el tiempo. Por tanto la simulación que predice los estados estacionarios que pueden ser alcanzados por el proceso se denomina simulación estacionaria.

- ✓ Simulación dinámica: Plantea los balances en su dependencia con el tiempo, ya sea para representar el comportamiento de equipos que utilizan procesamiento por lotes, o bien para analizar la evolución que se manifiesta entre dos estados estacionarios para un equipo o una planta completa. **(5)**

Clasificación desde el punto de vista de los procesos o sistemas que estudian

- ❖ Determinística: Se considera aquél en el cual las ecuaciones dependen de parámetros y variables conocidas con certeza, es decir que no existe incertidumbre ni leyes de probabilidades asociadas a las mismas. **(5)**

Para un mejor entendimiento si el proceso no contiene ningún elemento aleatorio, es decir, las relaciones funcionales entre las variables del sistema están perfectamente definidas y las variables de salida e internas quedan determinadas al especificar las variables de entrada, los parámetros y las variables de estado, estamos en presencia de una simulación determinística.

- ❖ Estocástica: Ocurre que ciertas variables estarán sujetas a incertidumbre, que podrá ser expresada por funciones de distribución de probabilidad. En este caso, por lo tanto, también los resultados del modelo estarán asociados a una ley de probabilidad. **(5)**

Modelo Matemático

En las ciencias experimentales y en las aplicaciones de las ciencias a problemas el especialista que analiza un fenómeno completo trata de sustituirlo por una estructura lógica que facilite su estudio. Esta estructura lógica se define por un conjunto de hipótesis sobre los elementos esenciales del fenómeno analizado y sobre las leyes que rigen su comportamiento. Generalmente el conjunto de hipótesis se plasma en la forma de un sistema matemático al que se denomina modelo. **(6)**

Un modelo es una representación de un objeto, idea, o sistema en una forma diferente a la entidad misma. En nuestro caso el modelo es un conjunto de relaciones matemáticas o lógicas derivadas de supuestos sobre el comportamiento del sistema. **(7)**

Como se puede ver el modelo matemático es una idealización abstracta del fenómeno real y es considerado el elemento fundamental de una simulación.

Simulación de Procesos

Se mencionó anteriormente que la simulación implica crear un modelo que se aproxime a un sistema del mundo real. Pero un modelo matemático por más detallado que sea no es el reflejo exacto de la realidad entre otras razones porque esta es demasiado compleja y variable para ser representada matemáticamente; al menos hasta estos momentos. Entonces surge la interrogante:

¿Cómo diseñar un modelo de un proceso continuo con materias primas de composición indeterminada y operaciones que no siempre se realizan de igual forma?

La solución es “Simular el Proceso” que se entiende como desarrollar modelos matemáticos que se concentren en los aspectos principales del proceso y que calculados repetida y creativamente permitan identificar las principales tendencias del comportamiento del proceso.

También la simulación de procesos es definida como una técnica para evaluar en forma rápida un proceso con base en una representación del mismo, mediante modelos matemáticos. **(1)**

Texson precisa que la simulación de procesos se utiliza generalmente para informar a las personas acerca de un proceso o un concepto que no se manifiesta visiblemente. Agrega además que en este tipo de simulación la persona escoge desde el principio los valores y parámetros y puede cambiarlos cuando lo desee hasta alcanzar mejores resultados. **(8)**

Pero con todo lo planteado anteriormente es evidente que para el estudio de la simulación es necesario tener bien claro la definición de proceso.

Un proceso químico es un conjunto de operaciones químicas y/o físicas ordenadas a la transformación de unas materias iníciales en productos finales diferentes. **(9)**

Desde otro punto de vista se puede decir que es considerado la serie de pasos o conjunto de operaciones que recorre un producto desde la adquisición de la materia prima hasta lograr el producto final.
Otra forma de definirlo es como un conjunto de actividades que se llevan a cabo dentro de un sistema.
¿Qué es un sistema?
Entiéndase sistema como una sección de la realidad compuesta por componentes que interactúan entre sí para lograr algún objetivo.
Si el simulador puede reemplazar al sistema real se denomina emulador.

Componentes Conceptuales

Todos los simuladores se diferencian por su arquitectura e implementación, pero todos tienen en común funcionalidad impuesta por las tareas que modelan. Esta funcionalidad se puede resumir en términos de 4 tipos de componentes conceptuales:

Programa ejecutivo*:* Este componente es el ejecutivo del simulador, el cual es responsable de la instalación de otros componentes y su colocación con el sistema operativo, interacciones con los usuarios, acceso y almacenamiento de datos, reporte y análisis de los cálculos de la simulación.

Operaciones unitarias*:* Estos componentes representan procesos físicos y también realizan papeles especializados tales como ejecución de cálculos adicionales para apoyar la optimización del proceso.

Propiedades físicas*:* Una funcionalidad crítica del simulador es la capacidad de modelar propiedades y el comportamiento de los materiales que son utilizados o creados por el proceso. Las propiedades y el comportamiento incluyen propiedades termodinámicas y de transporte.

Solucionadores numéricos*:* Esto incluye los métodos matemáticos especializados usados para evaluar las ecuaciones que describen una operación unitaria. **(10)**

Los modelos de operaciones unitarias, los componentes químicos y las propiedades físico-químicas constituyen la columna vertebral de los simuladores de procesos. La riqueza y efectividad de un simulador viene dada, entre otros aspectos, por la variedad, validez y alcance de su biblioteca de modelos, incluyendo las funciones de cálculo de las propiedades físicas de las corrientes.

DFP y DFI

Diagrama de Flujo del Proceso (DFP)

La Simulación de un proceso se inicia construyendo el Diagrama de Flujo de Materiales y Energía (DFP); que representa de forma esquemática los aspectos, de interés para la simulación, del flujo tecnológico y/o el sistema térmico de la fábrica. Está formado básicamente por equipos y corrientes, que "contienen" materiales y energía.

Diagrama de Flujo de Información (DFI)

El DFI es una representación gráfica del sentido en que fluye la Información inicial y la que se origina durante los cálculos. El DFI está formado por Módulos de Cálculo que representan matemáticamente lo que ocurre en los Equipos y/o Subprocesos y las Corrientes o Flujos de Informaciones que entran y salen de los Módulos. Como es de esperar para la confección de DFI se requiere del conocimiento del DFP.

Tipos de Simuladores de Procesos según la forma en que plantean su modelo matemático

Los simuladores de procesos pueden dividirse en los siguientes tipos según la filosofía bajo la cual se plantea el modelo matemático que representa el proceso a simular:

- Simuladores globales u orientados a ecuaciones
- Simuladores secuenciales modulares
- Simuladores híbridos o modular secuencial-simultáneo

En los simuladores globales se plantea el modelo matemático que representa al proceso construyendo un gran sistema de ecuaciones algebraicas que representa a todo el conjunto o planta a simular. De esta forma el problema se traduce en resolver un gran sistema de ecuaciones algebraicas, por lo general altamente no lineales.

Los simuladores modulares secuenciales se basan en módulos de simulación independientes que siguen aproximadamente la misma filosofía que las operaciones unitarias, es decir, cada equipo: bomba, válvula, intercambiadores, etc.; son modelados a través de modelos específicos para los mismos y además, el sentido de la información coincide con el "flujo físico" en la planta. **(5)**

Simulación a utilizar

Lo investigado hasta el momento ha servido de apoyo para decidir que se hará uso de la simulación estacionaria, determinística y modular secuencial para lograr el cumplimiento del objetivo de este trabajo.

Primeramente, de tipo estacionaria porque ayuda a comparar las distintas alternativas de condiciones de operación del proceso, con el fin de satisfacer las demandas pronosticadas. Por otra parte, permite que el proceso no sea demasiado sensible, esto es, que pequeños cambios en las entradas no provoquen grandes cambios en las salidas.

Se eligió además la simulación determinística debido a que ayudan a acercarse al conocimiento del comportamiento del proceso químico completo y sus cambios dinámicos. Además que el resultado de una sola simulación es una medida exacta del desempeño del modelo.

Por último se utilizará la simulación de procesos modular secuencial debido a que su filosofía tiene como ventaja una gran cantidad de características relevantes como su confidencialidad, robustez, es considerado poco versátil y muy flexible, es fácilmente comprendido por ingenieros que no son especialistas en simulación y la información ingresada por el usuario resulta fácil de chequear e interpretar.

Descripción General de los Simuladores de Procesos

Simulación y Optimización

En muchas ocasiones los términos optimización y simulación son causas de confusión. Lo primero a esclarecer es que las técnicas o métodos de optimización están diseñados para tomar la mejor decisión requiriendo la menor cantidad de experimentos. De ahí que optimización es determinar la acción que debe realizarse sobre el sistema para que este cumpla de la mejor manera posible con los objetivos deseados. **(11)** Pero generalmente los sistemas en estudio son tan complejos que la solución analítica no es posible. Por tanto la optimización debe llevarse a cabo a partir de resultados experimentales. Es aquí donde la simulación encuentra su primera aplicación ya que es capaz de proveer los datos necesarios.

Pasos para realizar un experimento de simulación

La simulación de procesos químicos está enmarcada por un conjunto de etapas que son las responsables de que los procesos de producción de la industria sean adecuadamente diseñados, operados y mejorados.

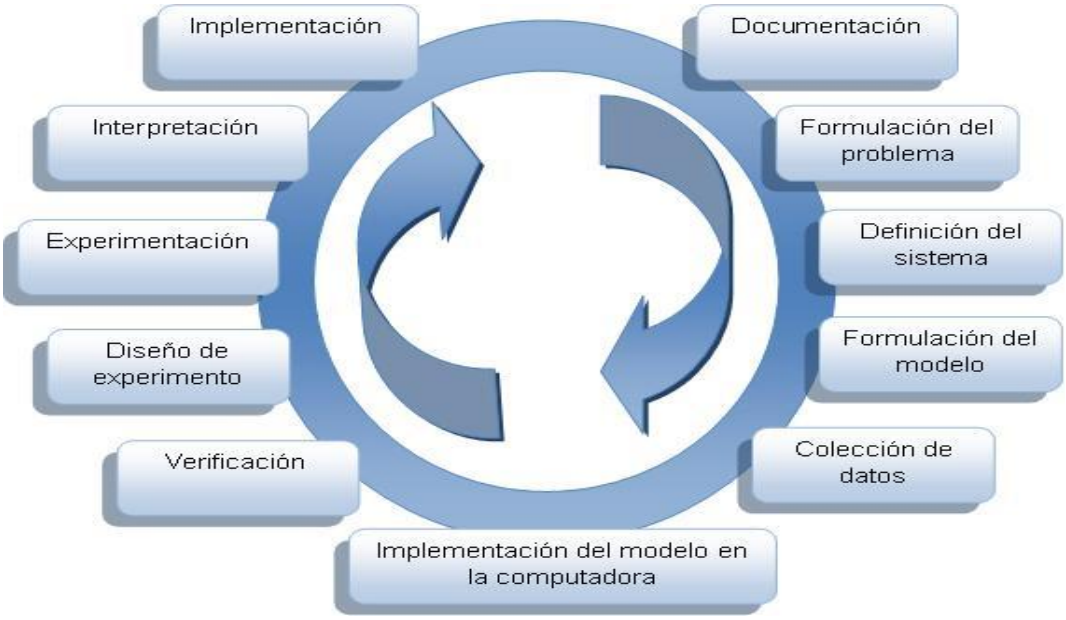

Ilustración 1 Etapas de un experimento de simulación

En el desarrollo de una simulación se pueden distinguir las siguientes etapas:

Formulación del problema: En este paso debe quedar bien claro el objetivo de la simulación. El cliente y el desarrollador deben acordar lo más detalladamente posible factores como: resultados que se esperan el plan de experimentación, el tiempo disponible, las variables de interés, el tipo de perturbación a estudiar, el tratamiento estadístico de los resultados y la complejidad de la interfaz del simulador y por último se debe establecer si el simulador será operado por el usuario o si solo recibirá resultados.

Definición del sistema: El sistema a simular debe estar perfectamente definido. Se debe acordar donde está la frontera del sistema a estudiar y las interacciones con el medio ambiente que serán consideradas.

Formulación del modelo: Se comienza con el desarrollo de un modelo simple que captura los aspectos relevantes del sistema real. La formulación del problema condicionará fuertemente esta etapa. El modelo simple se irá enriqueciendo como resultado de varias iteraciones.

Colección de datos: La naturaleza y calidad de datos están determinadas por la formulación del problema y el modelo. Los datos pueden ser provistos por: registros históricos, experimentos de laboratorios o mediciones realizadas en el sistema real. Los mismos deberán ser procesados adecuadamente para darles el formato exigido por el modelo.

Implementación del modelo en la computadora: El modelo es implementado utilizando algún lenguaje de programación. Existen lenguajes específicos de simulación que facilitan la tarea, también existen programas que ya cuentan con modelos implementados para casos especiales.

Verificación: En esta etapa se comprueba la exactitud del modelo desarrollado esto se lleva a cabo comparando las predicciones del modelo con las mediciones realizadas del sistema real, datos históricos o datos de sistemas similares.

Diseño de experimento: Aquí se definen las características de los experimentos a realizar: el tiempo de arranque, el tiempo de simulación y el número de simulaciones. No se bebe incluir aquí la elaboración del conjunto de alternativas a probar para seleccionar la mejor, esto pertenece a la optimización y no a la simulación.

Experimentación: Ya aquí corresponde realizar las simulaciones de acuerdo con el diseño previo. Los resultados obtenidos son debidamente recolectados y procesados.

Interpretación: Se analiza la sensibilidad del modelo con respecto a los parámetros que tienen asociados la mayor incertidumbre. El modelo será sensible a determinados parámetros si para pequeños cambios en los valores de los mismos las respuestas varían notablemente. Si es así se deberán recolectar datos adicionales para refinar la estimación de los parámetros críticos

Implementación: Conviene acompañar al cliente en la etapa de implementación para evitar el mal manejo del simulador o el mal empleo de los resultados del mismo.

Documentación: Incluye la elaboración de la documentación técnica y manuales de uso. La documentación técnica debe contar con la descripción detallada del modelo y de los datos, también debe incluir la evolución histórica de las distintas etapas de desarrollo. **(12)**

¿Cuándo utilizar la simulación?

No existe una formulación matemática: Muchos sistemas reales no pueden ser modelados matemáticamente con las herramientas actuales disponibles.

Existe una formulación matemática pero es difícil obtener una solución analítica: Los modelos matemáticos utilizados para modelar una planta química son imposibles de resolver en forma analítica sin realizar serias simplificaciones.

No existe el sistema real: Es el problema del ingeniero que tiene que diseñar un equipo nuevo o una nueva planta. El diseño del sistema mejorará notablemente si se cuenta con un modelo adecuado para realizar experimentos.

Los experimentos son imposibles debido a impedimentos económicos, de seguridad, de calidad o éticos: En este caso el sistema real está disponible para realizar experimentos pero la dificultad de los mismos hace que se descarte esta opción. Un ejemplo es la imposibilidad de provocar fallas en un avión real para evaluar la conducta del piloto.

El sistema evoluciona muy lentamente o muy rápidamente: Un ejemplo de dinámica lenta es el problema de los científicos que estudian la evolución del clima. Ellos deben predecir las conductas futuras del clima dado condiciones actuales, no pueden esperar que un tornado arrase una ciudad para luego dar el mensaje de alerta. Por el contrario existen fenómenos muy rápidos que deben ser simulados para poder observarlos con detalle, por ejemplo una explosión. **(11)**

Desventajas de la simulación

Aunque parece ser la herramienta ideal tienen que tenerse en cuenta sus desventajas:

El desarrollo de un modelo puede ser costoso laborioso y lento: El desarrollo de un simulador implica tiempo, fuerza y dinero. Cuando el sistema a simular existe siempre está presente la tentación de experimentar directamente con él en lugar de enfrentar su desarrollo.

Existe la posibilidad de cometer errores: Nunca se debe olvidar que la experimentación se lleva a cabo con un modelo y no con el sistema real, entonces si el modelo está mal o se cometen errores en su manejo, los resultados también serán incorrectos.

No se puede conocer el grado de imprecisión de los resultados: Por lo general el modelo se utiliza para experimentar situaciones nunca planteadas en el sistema real, por tanto, no existe información previa para estimar el grado de correspondencia entre la respuesta del modelo y la del sistema real. Este problema se puede atenuar de diversas formas. Cuando el sistema a simular no existe se puede recurrir a sistemas similares existentes para comparar su conducta con la predicha por el simulador. Cuando el sistema a simular existe pero nunca fue expuesto a las condiciones que se desean simular se puede evaluar la conducta del simulador estudiando condiciones históricas del sistema para las cuales si se conoce como respondió. **(11)**

Simuladores de procesos Internacionales

HYSYS

Desarrollado por Aspen Techonology, es un software que se ha especializado en la simulación de plantas petroquímicas. Entre sus características principales podemos mencionar:

- Es un programa interactivo orientado a objetos
- Interpretación de comandos de forma interactiva
- Capacidad de ejecutar todos los cálculos relacionados con las modificaciones en cualquier punto del proceso en forma bidireccional, tanto en procesos subsecuentes como predecesores.
- Reduce la necesidad de cálculos iterativos, lo que incide en su mayor velocidad de respuesta.

- Permite seleccionar el diagrama de flujo, es decir utilizar diferentes opciones de simulación.
- El ambiente de Hysys zcuenta con cuatro interfaces
    1. PFD (Process Flor Diagram): permite al usuario construir la topología del proceso que desea simular.
    2. Libro de Trabajo: colección de hojas de cálculo que despliegan la información de forma tabular.
    3. Propiedades: colección de páginas que contienen información acerca de los objetos que constituyen los procesos
    4. Resumen: lista de las corrientes y los módulos considerados. **(13)**

Las principales ventajas de HYSYS son:

- Su facilidad de uso con una interfaz amigable.
- Base de datos extensa.
- Utiliza datos experimentales para sus correlaciones.

Las principales desventajas de HYSYS son:

- Pocas o nulas aplicaciones de sólidos.
- Software de optimización limitado. **(14)**

Incluye herramientas para estimar:

- Propiedades físicas
- Equilibrios líquido vapor
- Balances de materias y energía
- Simulación de muchos equipos de ingeniería química.
- Simula procesos en estado estacionario y dinámico. **(15)**

SuperPro Designer.

Desarrollado por Simulation Sciences, es una herramienta de cálculo ambiental para el diseño integrado de procesos. La herramienta se orienta hacia la industria Bioquímica, Farmacéutica, de alimentos, así como para los procesos de disposición, reciclado y tratamiento de residuos. La herramienta se basa en el desarrollo de varias etapas que van desde las definición de los materiales a usar en el proceso, hasta la realización de reportes. Las fases que se desarrollan son las siguientes:

- Definición de componentes y mezclas.
- Clasificación y descripción de las corrientes.
- Descripción de los procedimientos, operaciones unitarias y equipos.
- Descripción de las condiciones de operación.
- Definición de las ayudas ofrecidas por la herramienta.
- Descripción de las características de los informes.
- Definición de las emisiones del proceso.
- Descripción de los casos de diseño.
- Evaluación de la interconectividad de la herramienta. **(16)**

CHEMCAD

Desarrollado por Chemstations Inc, es un simulador de procesos químicos con más de 40 años de historia y experiencia, es uno de los simuladores que se encuentra a la vanguardia hoy en día. CHEMCAD ha venido evolucionando durante estos años para convertirse en un paquete de módulos que abarca cálculo y diseño de intercambiadores de calor (CC-THERM), simulación de destilaciones dinámicas (CC-DCOLUMN), simulación de reactores por lotes (CC-ReACS), simulación de destilaciones por lotes (CC-BATCH), simulación de redes de tuberías (CC-SAFETY NET). **(17)**

Características Generales:

- Utilización de un solo modo para la elaboración de diagramas de flujo, la especificación, cálculo, y la creación de PFD
- Crea un solo archivo de simulaciones que son fáciles de trabajar y compartir
- Proporciona un acceso rápido y fácil a los datos de simulación a través del nuevo Explorador de CHEMCAD
- Los iconos de UNITOP diagrama de flujo y otras herramientas de dibujo siempre está disponible en el panel de la paleta personalizable
- Mensajes del panel de seguimiento de todos los mensajes de error y de advertencia, y puede almacenar notas de trabajo y mostrar los mensajes de seguimiento de ejecución
- CHEMCAD ahora ofrece espacio de trabajo simplificado de zoom y de centrado automático de diagramas de flujo de nuevos
- Elementos de la pantalla completamente personalizable incluir la clandestinidad, acoplamiento y desacoplamiento paneles, así como barras de herramientas personalizables
- Página principal con funciones de mostrar u ocultar cualquier elemento de diagrama de flujo individuo o grupo de elementos personalizados con un solo clic.

- Nueva estructura de la base de datos de componentes mejora la manipulación y la distribución de los componentes definidos por el usuario, más de 100 nuevos componentes añadidos.
- Auto-opción de convergencia disponible para las columnas de destilación
- VBA y VSTA son marcas registradas de Microsoft Corporation.
- Mejora el rendimiento y la interfaz de Administrador de licencias. **(18)**

Estas características se han mejorado sustancialmente, gracias a la compatibilidad que existe entre el Chemcad y el ConcepSys una potente herramienta de proceso, esta unión suma nuevas particulares al Chemcad como son:

- Simulaciones CHEMCAD pueden ser fácilmente importados en ConcepSys para continuar el trabajo de desarrollo del proyecto.
- Equipos y propiedades termodinámicas en el modelo de simulaciones CHEMCAD se utilizan como base para el dimensionamiento de equipos preliminares y el diseño 3D en ConcepSys.
- ConcepSys proporciona una visualización en 3D de simulación CHEMCAD
- Permite la generación de equipos para las hojas de proceso de las listas en línea, listas de equipo, dibujos, y las estimaciones de los costes necesarios para un paquete completo de proceso de diseño.
- El vínculo entre CHEMCAD y ConcepSys elimina los errores causados por la transferencia manual de los datos de simulación en herramientas de ingeniería preliminar.
- Los usuarios de CHEMCAD pueden ahora desarrollar una simulación, así como los documentos de diseño preliminar y las estimaciones de gastos por coeficientes de un flujo de trabajo de ingeniería suave y altamente productiva. **(19)**

Las compañías que patrocinan estos simuladores cuentan con el presupuesto y la tecnología para dar inmensos pasos en el desarrollo de los mismos, además cuentan con un factor fundamental la experiencia de muchos años desarrollando desde programas tradicionales escritos en FORTRAN hasta aplicaciones con tecnología orientada a objeto. Esta experiencia y los nuevos avances que se han logrando en el mercado de la simulación no están al alcance de las instituciones universitarias de países en vías de desarrollo como Cuba debido a que la accesibilidad está sujeta al pago de las respectivas licencias de uso.

Simulador de proceso nacional

STA (Sistema Termoazúcar)

Características Generales:

- La construcción del Diagrama de Flujo de Información (DFI), la creación de reportes, manejo y conversión de unidades de medidas de distintas magnitudes, las simulación matemática de los modelos, requisitos que son soportados en la arquitectura propuesta y da la medida de la adecuación del producto.
- Los ficheros que se guardan en la aplicación en un formato XML, este lenguaje de texto es compatible y manejado por la mayoría de las aplicaciones hoy en día.
- La posibilidad de exportar los diferentes diagramas a formatos de imágenes (jpg, bmp).
- Los reportes que realiza el simulador se pueden exportar a otros formatos como .pdf ó .doc
- Tiene definidos componentes que no permiten la entrada de cadenas no válidas, y en caso de que el usuario se valga de otros recursos para lograrlo, se le enseña a través de recursos visuales, donde está el error y no procesará la información hasta que esta sea entrada correctamente.
- El usuario es notificado en caso de cualquier error y se le da posibilidad de que consulte la ayuda relacionada con ese error. **(2)**

Caracterización la Arquitectura

El simulador STA se desarrollo sobre la plataforma .Net con una arquitectura en capas y orientada a componentes. Está compuesto por tres capas: Presentación, Negocio y Persistencia en las que se agrupan varios componentes que colaboran para cubrir las funcionalidades del simulador. Para la implementación de este simulador el sistema se estructuro en un grupo de librerías de vinculo dinámico (dll), conectadas a un ejecutable que forma el núcleo del simulador, esto permite la flexibilidad y la reusabilidad esto permite la creación de nuevos simuladores modulares secuenciales con características semejantes, con tiempos de desarrollado relativamente cortos y sin cambios significativos en la misma. Para este simulador la arquitectura esta divida por 3 capas:

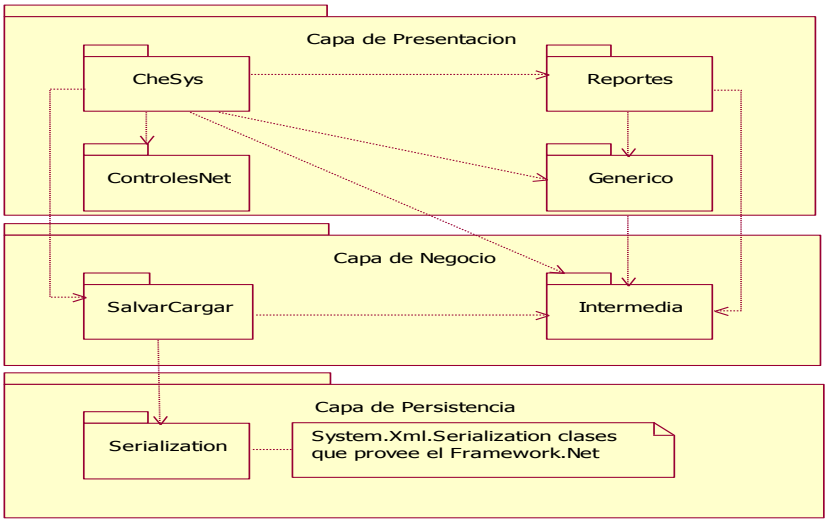

Ilustración 2 Arquitectura del STA (20)

Capa de Presentación

CheSys (.exe): funciona como el agente integrador, en el se encuentran los elementos gráficos del simulador y la posibilidad de acceder a funcionalidades adicionales.

Reportes (.dll): Este componente facilita el manejo del flujo de reportes que generan en la simulación, para su uso se utilizo la herramienta CrystalReports, además permite personalizar los reportes los mismos pueden ser confeccionados específicos o especializados.

Genérico (.dll): componente que contiene un conjunto de clases que garantizan el funcionamiento del editor grafico, la herramienta utilizada fue la plataforma .NET, la función principal de este paquete es la encapsulación de funcionalidades de los elementos visuales de la aplicación.

Capa de Negocio

SalvarCargar (.dll): este paquete permite el trabajo con los ficheros y la serialización de objetos, para el mismo se utilizo las ventajas que ofrece la plataforma .NET, de esta forma se puede salvar y cargar los proyectos sin pérdida de información.

Intermedia (.dll): este componente provee los mecanismos para el cálculo de las propiedades de módulos y corrientes. Provee un paquete de funciones para el cálculo de propiedades físico-químicas generales y para la conversión de unidades.

Capa de Persistencia

System. Xml. Serialization clases que provee el marco de trabajo .Net.

El uso de otros componentes

En estas capas solo se mencionan los componentes principales que constituye el núcleo principal, este brinda funcionalidades generales y mecanismos genéricos para la implantación de los módulos de cálculo, estos se implementan como bibliotecas de vinculo dinámico independientes y se integran posteriormente a la aplicación principal como la comunicación de ellos no está preestablecida, es decir, de inicio no se sabe que componentes se van a relacionar en un determinado proyecto de simulación, todas las interfaces fueron definidas en Intermedia, entonces los módulos implementan las interfaces según las características del equipo real de la industria. Entonces dos componentes se pueden comunicar si implementan la misma interfaz.

Gestión de operaciones unitarias, componentes y propiedades físico-químicas.

Juega un papel protagónico en la parte del negocio, este componente provee los mecanismos para el cálculo de las propiedades de módulos y corrientes. Provee un paquete de funciones para el cálculo de propiedades físico-químicas generales y para la conversión de unidades. Aquí se definen las interfaces que rigen la comunicación entre los diferentes módulos de cálculo así como los tipos de corrientes que puede manejar el simulador, estas corrientes están estrechamente ligadas a la tecnología que se esté simulando, en la actualidad solo están definidas las corrientes del proceso de obtención y refinación de azúcar. Este componente provee la estructura general de un módulo y sus puertos desde el punto de vista del negocio, entonces un módulo determinado se especificara en un componente independiente, el cual contendrá una clase que es una especificación de la clase módulo que está en Intermedia. Dentro de Intermedia se encuentran submódulos que encapsulan funcionalidades afines, los paquetes General, Negocio, Interfaces y Parámetros constituyen estos submódulos. **(20)**

Operaciones unitarias.

La simulación consiste en la elaboración de modelos matemáticos de los equipos, representando una aproximación de su comportamiento. Dichos modelos son implementados en la clase NegModulo. Esta clase tiene una serie de métodos y propiedades entre los que se encuentra Calcular, en el que se llama el método CalcularModulo (en el que se programan los modelos) y manda a cada uno de los puertos de salida a pasar su valor al puerto con el que está conectado. Con este mecanismo se garantiza que cada módulo que representa un equipo del proceso real realice sus operaciones de manera independiente y una vez que haya calculado entregue los datos obtenidos a la siguiente unidad de operación unitaria.

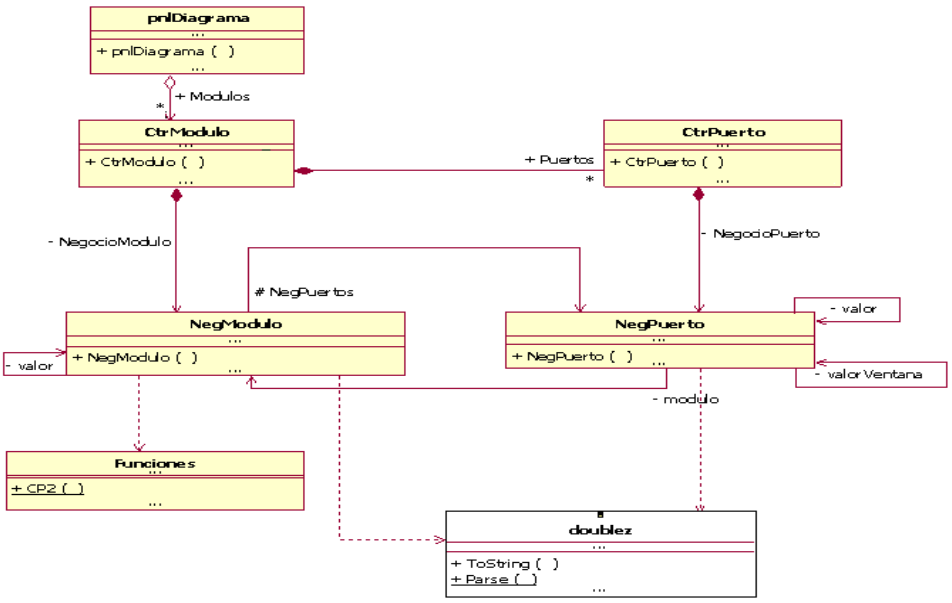

Ilustración 3 Realización del CU Simular en el STA (20)

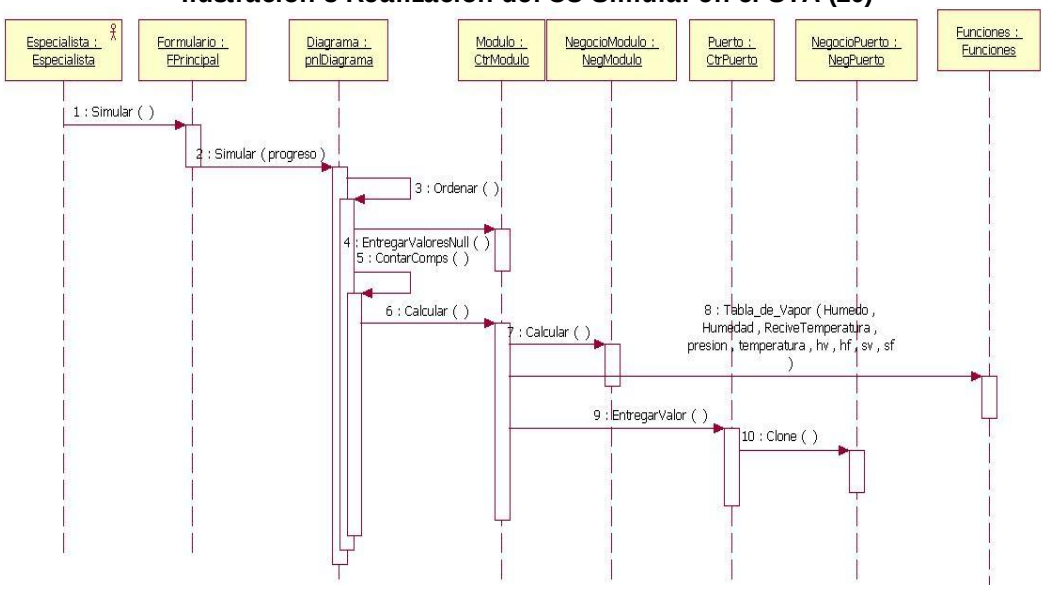

Ilustración 4 Realización del CU Simular en el STA (20)

Corrientes y componentes.

Los tipos de corrientes que aparecen en el simulador vienen dadas por la tecnología que se esté simulando, en la primera etapa de desarrollo se incluyen los modelos de la industria azucarera, las muchas corrientes que existen en esta industria se pueden clasificar en dos grandes familias, flujos azucarados y no azucarados, la diferencia está en la cantidad de parámetros que contiene la corriente. Ante esta situación se definieron dos clases que heredan de NegPuerto, las cuales tienen ya establecidos los parámetros que lleva cada tipo de corriente, estas fueron NegPuertoAgua y NegPuertoAzucarado. De

esta manera la creación de cualquier flujo del proceso de producción de azúcar se obtiene heredando de una de estas dos clases.

En el paquete Intermedia se definen todas las interfaces, la cuales dan las operaciones para acceder a los diferentes componentes de un corriente. En la industria azucarera, por ejemplo, existen dos tipos de flujos: flujos azucarados y flujos de agua-vapor, que se dividen en estos dos grupos por los componentes que las integran. De esta manera un módulo que procese jugo por ejemplo, va a implementar la interfaz IFluJugo, y así todos los módulos que procesen jugo se pueden ser conectados e intercambiar información. **(20)**

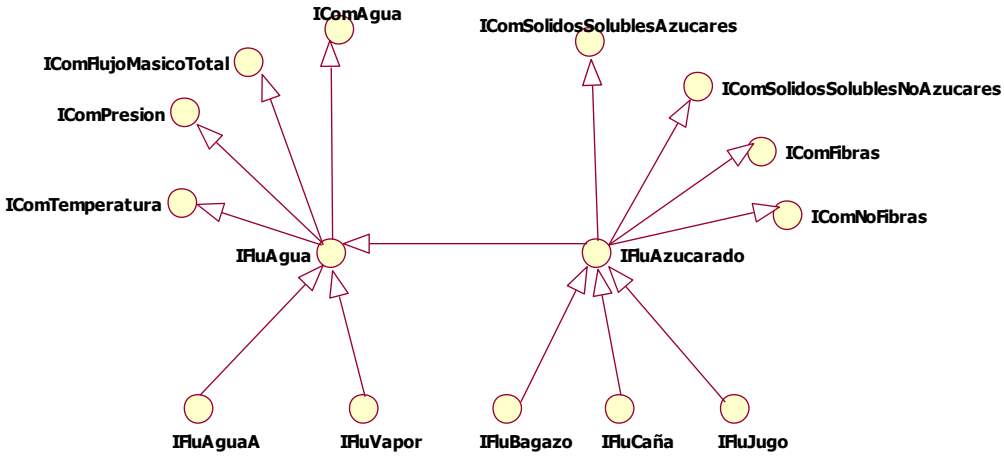

Ilustración 5 Interfaces del STA (20)

Evaluación de propiedades.

La evaluación de propiedades en el STA se hace mediante la clase Funciones del paquete Intermedia. En esta clase se implementan métodos para evaluar las propiedades físicas y químicas de los componentes de las corrientes involucradas en los procesos y tecnologías que se simulan.

Tabla 1 Evaluación de propiedades en el STA (20)

Nombre: Funciones		Tipo: Clase
Descripción: Contiene todos la funciones implementadas que son comunes a los módulos, y que su utilizan el cálculo de propiedades químicas.		
Atributo	Tipo	Visibilidad
-	-	-
Métodos y propiedades		
Nombre	Tipo de retorno	Visibilidad
CP2()	double	public
Descripción: Calcula de capacidad calorífica de las corrientes azucaradas.		

HJ2()	double	public
Descripción: Calcula de Entalpía específica de las corrientes azucaradas.		
Psat2()	double	public
Descripción: Calcula la Presión de Saturación		
EPEB()	double	public
Descripción: Calcula de la elevación del punto de ebullición		
TempSaturacion()	double	public
Descripción: Calculo de los Temperatura de saturación		
EntalpiaLiquidoSaturado()	double	public
Descripción: Calcula la entalpía de liquido saturado(solo para el agua)		
EntropiaLiquidoSaturado()	double	public
Descripción: Calculo de la entropía de liquido saturado(solo para el agua)		
EntalpiaEspecificaVaporSat()	double	public
Descripción: Calcula de la entalpía del vapor saturado(solo para el agua)		
EntropiaEspecificaVaporSat()	double	public
Descripción: Calcula la entropía del vapor saturado(solo para el agua)		
EntalpiaEspecificaVaporSobreCal()	double	public
Descripción: Calculo de la entalpía del vapor sobrecalentado(solo para el agua)		
EntropiaEspecificaVaporSobreCal()	double	public
Descripción: Calculo de la entropía del vapor sobrecalentado(solo para el agua)		
Stabl2()	void	public
Descripción: Steamtables 2		
Pureza()	double	public
Descripción: Calcula de la Pureza		
Brix()	double	public
Descripción: Calculo del Brix		

Deficiencias del STA

- El uso de Visual Studio una herramienta que no es multiplataforma y es software propietario, se recomendaría el uso del lenguaje Java.

- El tratamiento de las corrientes como una interfaz y no como una clase, impide que las mismas se puedan gestionar, esto incide en una violación del patrón experto, es decir, debería existir una clase corriente encargada de gestionar todo lo referente a ella.
- Los métodos de las propiedades están implementadas en el paquete Funciones de Intermedia y no existe la clase propiedades, aquí se violan dos patrones el experto porque no existe la clase propiedades que gestione todos sus métodos, así como el de alta cohesión porque la clase Funciones está sobrecargada con métodos con propósitos diferentes.
- El cálculo de las operaciones unitarias no es el más óptimo, para el mismo se recomendaría el tratamiento de la operaciones unitarias como un grafo, de esta manera se podría usar para el recorrido eficientes algoritmos existentes.

Metodología y Herramientas

Metodología de Desarrollo de Software

Las metodologías tradicionales fueron creadas entre los años 70 y 80 con el fin de ayudar a los profesionales indicando pautas para realizar y documentar cada una de las tareas del desarrollo del software. Sin embargo, casi todas tienen un gran problema: los pasos que sugieren para llevar a cabo cada tarea están cargados de reiteraciones y ambigüedades. No suelen tener en cuenta cosas como la calidad, la satisfacción, la competitividad y los beneficios.

Entre las metodologías tradicionales podemos citar:

- Desarrollo de sistemas de Jackson (JSD).
- Ingeniería de la información.
- Structured System Analysis and Design Method (SSADM).
- MÉTRICA

Las metodologías surgidas desde los 90 hasta aquí suelen tener una cierta agilidad**.** Siendo conscientes de lo cambiante y amplio que es el mundo del software, una metodología debe ser lo suficientemente precisa como para que todo el mundo la pueda seguir y utilizar. También debe ser lo suficientemente adaptable como para poder aplicarse en distintos proyectos y lo suficientemente sencilla como para que no resulte muy tediosa su utilización, así como, lo suficientemente completa y compleja como para que la utilización por parte del equipo sea provechosa.

Entre las metodologías modernas**:**

- Rapid Application Development (Desarrollo rápido de aplicaciones - RAD)
- Scrum
- Extreme programming. (Programación extrema - XP)
- Rational Unified Process. (Proceso Racional Unificado - RUP)
- Agile Unified Process. (Proceso Ágil Unificado - AUP) **(21)**

Criterios de selección de las metodologías ágiles

- Son iterativas. No intentan minimizar los cambios, sino estar preparados para aceptarlos.
- Son incrementales. Posee ciclos rápidos.
- Son adaptativas en lugar de repetibles. Es posible realizar cambios de último momento.
- Son cooperativas. Clientes y desarrolladores trabajan juntos.
- Priorizan a los individuos y a las interacciones por sobre los procesos.
- Incorporan la retroalimentación sobre el proceso. Priorizan la colaboración con el cliente.
- Prefieren código ejecutable antes que una documentación exhaustiva.

Metodología seleccionada

Agile UP (AUP) es una versión simplificada de Rational Unified Process (RUP) que describe simple y fácil el enfoque de desarrollo de software utilizando técnicas y conceptos de agilidad y aún manteniéndose fiel al RUP. Su ciclo de vida es serial en lo grande e iterativo en lo pequeño, liberando entregables incrementales en el tiempo. Surge en septiembre del 2005 y su primera versión el 13 de mayo del 2006.

¿Por qué AUP?

- Los requerimientos son altamente volátiles.
- El cliente entiende el proceso y está involucrado en el proyecto.
- Se cuenta con profesionales capacitados y competentes.
- Se tienen canales ricos de comunicación.
- El grupo de trabajo no es demasiado grande. (~ 50)
- Se desea fomentar la mejora continua del proceso. **(22)**

A todo lo mencionado anteriormente hay que agregar que fue elegida principalmente porque se encuentra entre XP y lo tradicional de RUP. Es un proceso con técnicas ágiles manteniendo formalidades de RUP, siendo esta última la metodología estudiada en la carrera y la más utilizada en la universidad.

El ciclo de vida de AUP

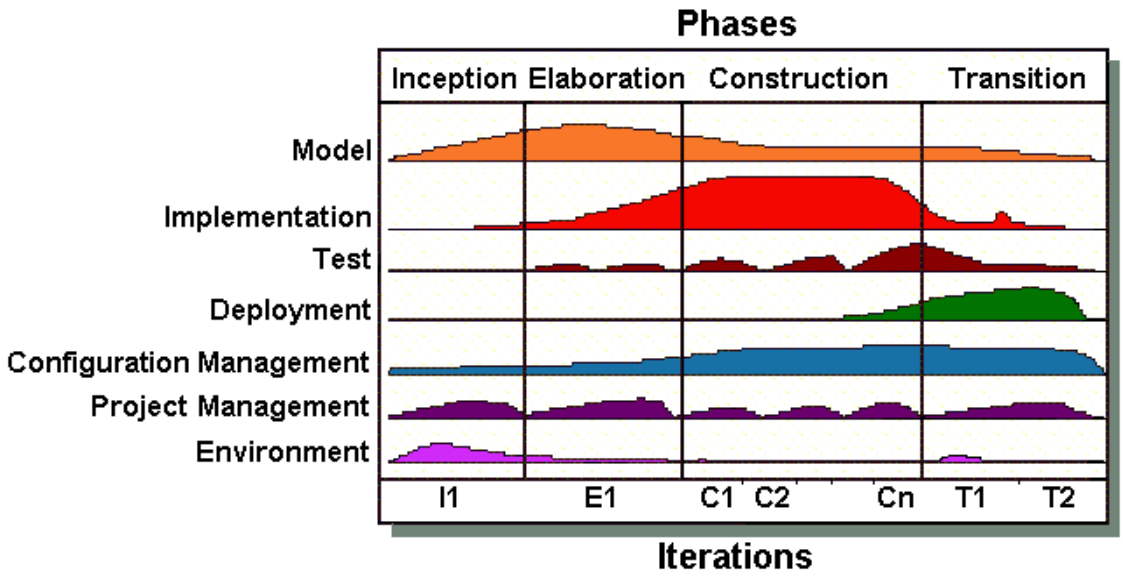

Ilustración 6 Ciclo de vida de AUP (23)

La naturaleza de serie de Agile UP es capturado en sus cuatro fases:

1- Inicio: El objetivo es identificar el alcance inicial de proyecto, una arquitectura inicial del sistema y obtener un presupuesto inicial del proyecto y una aceptación de los involucrados.

2- Elaboración: El objetivo es probar la arquitectura del sistema.

3- Construcción: El objetivo es construir software que trabaje sobre una base regular, incremental, que se reúne la más alta prioridad a las necesidades del proyecto los interesados.

4- Transición: El objetivo es validar y desplegar el sistema en su ambiente de producción.

Flujos de trabajo de Agile UP

Los flujos de trabajo son ejecutados en una forma iterativa, definiendo las actividades que el equipo de desarrollo ejecuta para construir, validar y liberar software funcional, el cual cumple con las necesidades del usuario.

- Modelado: La meta de este flujo es entender el negocio de la organización, el dominio del problema que el proyecto aborda e identificar una solución viable para abordar el dominio del problema.
- Implementación: La meta de este flujo es transformar su modelo(s) en un código ejecutable y realizar una prueba de nivel básico en una unidad particular de prueba.
- Pruebas: La meta de este flujo es ejecutar una evaluación de los objetivos para asegurar la calidad. Esto incluye encontrar defectos, validar que el sistema función como fue diseñado y verificar que los requerimientos están completos.

- Despliegue: La meta de ésta disciplina es planificar la entrega del sistema y ejecutar el plan para que el sistema esté disponible para los usuarios finales.
- Administración de la Configuración: La meta de de este flujo es administrar el acceso a los entregables o productos del proyecto. Esto incluye no sólo el rastreo de versiones del producto en el tiempo, sino que también incluye controlar y administra los cambios que ocurran.
- Administración del Proyecto: La meta de de este flujo es dirigir las actividades que se llevan a cabo en el proyecto. Esto incluye administración del riesgo, la dirección de personas (asignar tareas, seguimiento de los procesos, etc.), y coordinar con los sistemas y personas fuera del alcance del proyecto para que el este termine a tiempo y dentro del presupuesto.
- Entorno: La meta de este flujo es apoyar el resto de los esfuerzos por garantizar que, el proceso adecuado, la orientación (normas y directrices) y herramientas (hardware, software, etc.) estén disponibles para el equipo según sea necesario. **(23)**

Distinciones de AUP

1. Mantener los productos de trabajos tan simples y concisos como sea posible.
2. Se necesita menos documentación que con la metodología RUP.
3. El trabajo se debe realizar cerca de las personas con las cuales se está creando el producto para hacer sólo lo que ellos necesitan.
4. Los documentos ágiles son los suficientemente buenos para la tarea en cuestión.
5. Producir un documento es la peor forma de comunicar información, muchas personas paradas frente a una pizarra hablando es la mejor manera.
6. Usar herramientas simples como una pizarra (antigua) y hojas de papel para modelar y capturar documentación.
7. Considerar adoptar las plantillas de código abierto como base de partida para crear su propio trabajo.

Lenguaje de Programación

La principal característica de los lenguajes de programación es que ofrecen una librería de instrucciones suficientemente completa como para permitir programar cualquier tipo de modelo por complejo que este sea.

En la simulación cuando llega el momento de describir el modelo en un lenguaje se tienen dos cursos de acción a seguir:

- Desarrollar el software requerido.
- Comprar software (lenguajes de programación de propósito especial)

Ejemplos: GPSS, GPSSH, PROMODEL SIMFACTORY, SLAM, MICROMANAGER, etc.

Cuba desde el año 2005 tiene como meta emigrar hacia el software libre con el objetivo de alcanzar soberanía y autonomía tecnológica. Por esta razón es que en esta investigación se seleccionó como lenguaje Java.

Criterios que se tuvieron en cuenta para seleccionar Java

- Tiene estandarizado una interfaz orientada a objetos para acceder de un modo portable a cualquier base de datos. Esta API se denomina Java Database Connectivity.
- Es poseedor de una máxima portabilidad ya que un programa compilado de java puede ser interpretado en cualquier computadora que tenga implementado el intérprete de java.
- Es un lenguaje robusto ya que maneja la memoria de la computadora de tal manera que el programador no tiene que preocuparse por eso y además verifica su código al mismo tiempo que lo escribe, y una vez más antes de ejecutarse, de manera que se consigue un alto margen de codificación sin errores.
- Es fácil de comprender y utilizar por la simple razón de que es más complejo que un lenguaje simple, pero más sencillo que cualquier otro entorno de programación.
- Tiene la política de estandarización de librerías brindando una manera clara y segura de diseñar y codificar a partir de un conjunto de tecnologías orientadas hacia distintos tipos de aplicaciones:

    - Java SE (Java Standard Edition) considera facilidades de propósito general y aplicaciones de escritorio en particular.
    - Java EE (Java Enterprise Edition) para el desarrollo de aplicaciones empresariales (potencialmente sofisticadas) en servidores.
    - Java ME (Java Micro Edition) dirigida a la programación de dispositivos móviles.

- Tiene licencia GNU GPL desde finales del 2007 de tal forma que prácticamente todo el Java de Sun es ahora software libre (aunque la biblioteca de clases de Sun que se requiere para ejecutar los programas Java aún no lo es).
- Los niveles de seguridad que presenta son:

- Fuertes restricciones al acceso a memoria, como son la eliminación de punteros aritméticos y de operadores ilegales de transmisión.
- Rutina de verificación de los códigos de byte que asegura que no se viole ninguna construcción del lenguaje.
- Verificación del nombre de clase y de restricciones de acceso durante la carga.
- Sistema de seguridad de la interfaz que refuerza las medidas de seguridad en muchos niveles. **(24)**

¿Por qué se utilizó Eclipse como IDE?

¿Qué es un IDE?

Entorno de Desarrollo Integrado

Son un conjunto de herramientas para el programador, que incluyen un buen editor de código, administrador de proyectos y archivos, enlace transparente a compiladores e integración con repositorios.

Ejemplos de IDEs en Java

AnyJ, NetBeans, Sun Java Studio Creator, Borland JBuilder, IBM WebSphere Studio Application Developer, Sun One Studio y Eclipse.

La mayoría de estos IDE's son propietarios de compañías como Sun, Borland o IBM lo cual representa un inconveniente monetario si quieres operar con una licencia legal.

¿Qué es Eclipse?

Eclipse es un proyecto de desarrollo de software de código fuente abierto (open source) y basada en módulos conectables (plug-in) cuyo objetivo es la construcción de herramientas integradas para el desarrollo de aplicaciones.

Un plug-in es la mínima unidad de la plataforma que puede ser desarrollado por separado y que le aporta una nueva funcionalidad. **(25)**

Criterios de selección de Eclipse versión 3.4

- ✓ Es una plataforma de desarrollo libre y de código abierto basada en java.
- ✓ Soporta múltiples lenguajes es decir varias tecnologías asociadas.
- ✓ Editor visual con sintaxis coloreada.
- ✓ Compilación incremental de código.
- ✓ Modifica e inspecciona valores de variables.
- ✓ Avisa de los errores cometidos mediante una ventana secundaria.
- ✓ Depura código que resida en una máquina remota.

- ✓ ECLIPSE es soportado por los principales sistemas operativos:
  - Linux
  - Windows
  - Solaris 8 (SPARC/GTK 2)
  - Mac OSX –Mac/Carbon **(25)**

Lenguaje Unificado de Modelado (UML) versión 2.0

El Lenguaje Unificado de Modelado (UML) es un lenguaje para la especificación, la visualización, la construcción y la documentación de los artefactos de los sistemas de software y también para otros tipos de sistemas. Representa una colección de las mejores prácticas de ingeniería que han sido probadas con éxito en el modelado de sistemas grandes y complejos. **(26)**

La principal ventaja de UML es ser un lenguaje de propósito general, aunque esto en ocasiones se puede convertir en una desventaja, porque no se pueden representar cabalmente las situaciones o características propias de dominios específicos. **(27)**

¿Por qué se utilizó Visual Paradigm como herramienta CASE?

¿Qué es Case?

(Ingeniería de Software Asistida por Computadora) Enmarca diferentes tipos de programas que se utilizan para ayudar a las actividades de proceso del software. También incluyan un generador de código fuente automático que trabaja a partir del modelo del sistema y de algunas guías de procesos para los ingenieros del software. **(28)**

Visual Paradigm versión 6.4

Es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Este software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. **(29)**

Criterios de Selección de Visual Paradigm

- ✓ Es código abierto, soporta la versión 2.0 de UML.
- ✓ Permite su extensión mediante la conexión de módulos conectables (plug-in) o usando plantillas (templates).
- ✓ Es una herramienta UML fácil de usar

- ✓ Ingeniería inversa.
- ✓ Generación de código
- ✓ Importación desde Rational Rose
- ✓ Exportación/importación XMI
- ✓ Generador de informes.
- ✓ Editor de figuras.
- ✓ Integración con MS Visio, plug-in.
- ✓ Integración IDE con Visual Studio, IntelliJ IDEA, Eclipse, NetBeans.**(30)**

Pero aunque indiscutiblemente Visual Paradigm es una herramienta potente la razón fundamental de su elección es que en la Universidad de Ciencias Informáticas se paga la licencia de uso, de ahí su alta aceptación.

Conclusiones

La investigación realizada en este capítulo brindó la información necesaria acerca de la simulación de procesos que influyó en la decisión de utilizar la simulación estacionaria, determinística y modular secuencial para el cumplimiento de los objetivos propuestos. La caracterización de simuladores internacionales aunque con información limitada por causa de la poca accesibilidad mostró nuevas prestaciones y servicios que utilizan estos programas. El STA (Sistema Termoazúcar) sirvió para conocer a profundidad todo lo referente a cómo desarrollar y mejorar su subsistema de componentes químicos, propiedades físico-químicas y modelos de operaciones unitarias. Las características del proyecto fueron determinantes para elegir como metodología a utilizar la Agile UP. Para el lenguaje y las herramientas se tuvo en cuenta principalmente que fueran libres y de código abierto.

CAPÍTULO 2: Presentación de la solución propuesta.

Introducción

En este capítulo se realiza la presentación de la solución propuesta con el objetivo de entender el problema en la presente investigación. Primeramente se expone el diagrama de clases del modelo del dominio con la definición de cada una de sus clases. Luego se mencionan los requisitos funcionales y no funcionales agrupándolos en casos de usos del sistema. Por último con los casos de usos del sistema y sus descripciones se realiza un diagrama de casos de usos del sistema.

Modelo del Dominio o Conceptual

El Modelo de Dominio es una representación visual estática que captura los tipos más importantes de objetos o eventos que suceden en el entorno real donde está el sistema. Este modelo se describe mediante diagramas de UML y tienen como objetivo entender las principales clases dentro de lo que abarca el sistema posibilitando a los involucrados en el desarrollo del producto ya sean clientes, desarrolladores e interesados utilizar un vocabulario común, el cual permita capturar correctamente los requisitos y por ende obtener el producto deseado. **(26)**

¿Por qué modelo del dominio?

El modelo del dominio se realiza cuando no se logra determinar el proceso del negocio con fronteras bien establecidas, es decir, no se logra definir claramente quienes son las personas que lo inician, quienes son los beneficiados con cada uno de estos procesos, pero además quienes son las personas que desarrollan las actividades en cada uno de los procesos.

Definición de las clases del modelo del dominio

Proceso Químico: Es el conjunto de operaciones químicas que recorre un producto desde la adquisición de la materia prima hasta lograr el producto final.
Corriente: Se refiere a los flujos de distintas sustancias que existen en la industria química.
Componente Químico: Sustancia química involucrada en el proceso.
Propiedades Físico- Químicas: Conjunto de propiedades ya sean físicas o químicas que representan las características de un componente químico determinado.
Propiedades de equipo: Conjunto de propiedades que representan características de un equipo de terminado.

Magnitud: Propiedad asociada a un fenómeno que se puede medir y realizar criterios de igualdad y suma y que además posee factores de conversión de unidades.

Modelo Matemático: Sistema de ecuaciones que representan el funcionamiento de un fenómeno real, que con su resolución repetitiva y creativa, se puede obtener una predicción de su comportamiento.

Resultado: Es el conjunto de información resultante de la simulación de procesos que luego se utiliza en la optimización y análisis.

Simulación: Es una técnica matemática que imita el funcionamiento de un proceso del mundo real en un tiempo determinado, con el propósito de explicar, entender o mejorar el proceso.
Operación Unitaria: Representan los equipos que son usados en los distintos procesos.

Diagrama de clases del Modelo del Dominio

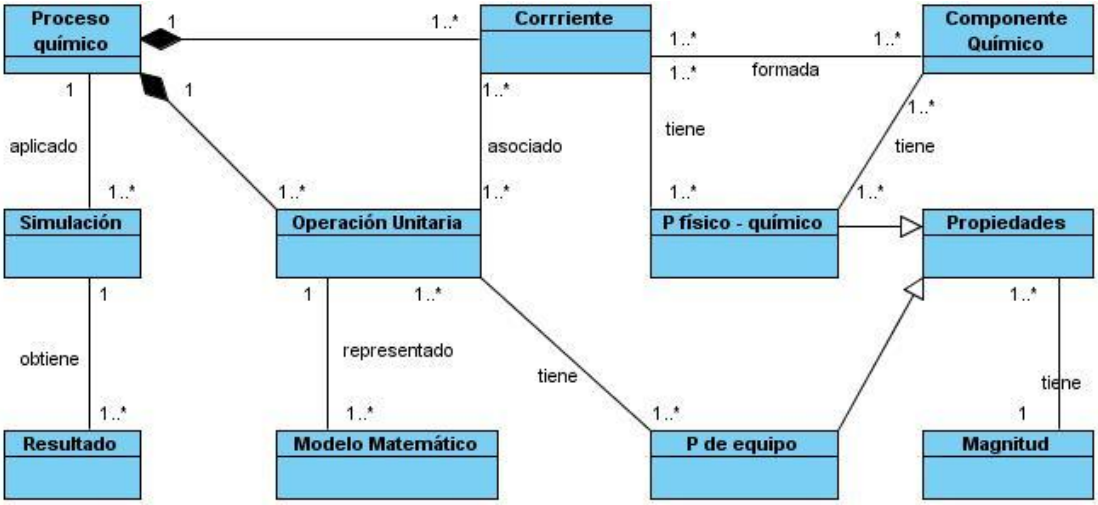

Ilustración 7 Diagrama de clases del modelo del dominio

Descripción del Modelo del Dominio

En este diagrama se puede observar que a un Proceso químico se le es aplicado una o varias simulaciones con el objetivo de obtener resultados útiles para un posterior análisis. Un Proceso químico está compuesto por Corrientes y Operaciones Unitarias. Una Operación Unitaria se representa por al menos un Modelo Matemático. Existen dos tipos de propiedades las de equipo que las presentan las Operaciones Unitarias y las propiedades físico- químicas que tienen las Corrientes y los Componentes Químicos. De las Corrientes se puede describir que están formadas por varios Componentes Químicos y

asociadas a como mínimo una Operación Unitaria. Finalmente las propiedades en general tienen una magnitud para su medición.

Requerimientos

Los requerimientos son capacidades y cualidades que necesita un usuario o cliente que sean satisfechas para resolver un problema o lograr un objetivo específico. Existen dos tipos de requerimientos los funcionales y los no funcionales.

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir.

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable.

Para la recogida de la lista de los requerimientos que debe poseer el subsistema a desarrollar se utilizó como técnica de captura de requisitos la lluvia de ideas por ser una técnica de grupo que permite la obtención de un gran número de ideas sobre un determinado tema de estudio.

Requerimientos Funcionales

RF.1 Gestionar Proceso.

- RF.1.1 Listar corrientes.
- RF.1.2 Crear corriente.
- RF.1.3 Eliminar corriente.
- RF.1.4 Listar operaciones unitarias.
- RF.1.5 Adicionar operaciones unitarias.
- RF.1.6 Eliminar operación unitaria.

El subsistema debe permitir gestionar un proceso, para el mismo se debe tener la capacidad de devolver una lista de corrientes y operaciones unitarias, crear corrientes y operaciones unitarias así como debe brindar la funcionalidad de eliminar una corriente o una operación unitaria.

RF.2 Obtener propiedades de la corriente.

El subsistema permitirá obtener las propiedades de una corriente a partir de los datos introducidos.

RF.3 Obtener propiedades de operación unitaria.

El subsistema permitirá obtener las propiedades de una operación unitaria a partir de los datos introducidos.

RF.4 Calcular operación unitaria.

- RF.4.1 Calcular parámetros de corrientes.

RF.4.2 Calcular parámetros de operación unitaria.

RF.4.3 Modificar propiedades específicas de la operación unitaria.

El subsistema permitirá el cálculo de una operación unitaria, para realizarlo se deberá contar la capacidad de calcular los parámetros de las corrientes y de las operaciones unitarias, además de modificar las propiedades específicas de la operación unitaria.

RF.5 Evaluar propiedades.

RF.5.1 Calcular propiedades de corriente.

RF.5.2 Calcular propiedades de componente.

El subsistema a partir de la presión, temperatura y composición de la corriente que ha sido modificada, deberá permitir calcular las propiedades de los componentes asociados a dicha corriente y calcular además las propiedades dependientes de la corriente modificada. A partir de lo cual el sistema será capaz de modificar las propiedades de la corriente sustituyéndolas por las calculadas.

RF.6 Identificar corrientes.

El subsistema debe ser capaz de identificar las corrientes del proceso.

RF.7 Identificar operación unitaria.

El subsistema debe ser capaz de identificar las operaciones unitarias.

RF.8 Introducir corriente de entrada.

RF.8.2 Modificar propiedades de la corriente.

El subsistema permitirá introducir una corriente de entrada así como modificar las propiedades de la misma.

RF.9 Simular proceso.

El subsistema deber permitir simular un proceso, para el mismo primero se deberá calcular el orden de cálculo y después proseguir en el orden determinado el cálculo de cada operación unitaria que se encuentre en el proceso.

RF.10 Determinar orden de cálculo.

El subsistema deberá ser capaz de determinar el orden de cálculo de las operaciones unitarias.

RF.11 Introducir propiedades de operación unitaria.

Le subsistema deberá permitir introducir las propiedades de las operaciones unitarias.

RF.12 Manipular unidad de medida.

El subsistema debe ser capaz de realizar las transformaciones correspondientes en las unidades de medida y modificar los valores, además de las unidades de medida.

RF.13 Salvar Resultado.

El subsistema debe ser capaz de salvar la información en ficheros para su posterior análisis.

Requerimientos No Funcionales

- ✓ **Requerimientos de Software:**

El subsistema requiere de la instalación de la máquina virtual de Java versión 5.0.

- ✓ **Requerimientos de Hardware:**

Para la elección de los requerimientos de hardware fueron tomados los sistemas operativos actualmente más usados: Windows, GNU Linux y Mac OS X v10.4.

Capacidad: Mínimo 3.5Gb
RAM: Mínimo 256 Mb (Recomendado 512 o más)
Procesador: Mínimo 867 MHz

- ✓ **Restricciones en el diseño y la implementación:**

Lenguaje de programación a ser usado Java.

Eclipse versión 3.3 como herramienta IDE.

Visual Paradigm versión 6.4 como herramienta CASE para el uso de UML como lenguaje de modelado.

Debe ser una aplicación de escritorio.

- ✓ **Requerimientos de apariencia o interfaz externa:**

El subsistema de modelos, componentes químicos y propiedades no interactúa directamente con el usuario de la aplicación, por tanto no se tienen requisitos de apariencia o interfaz externa.

Descripción del Sistema Propuesto

Descripción de los actores

Los actores representan un rol externo que interactúa con el sistema y es llevado a cabo por una persona, hardware o sistema. Se identifican como usuarios que interactúan directamente con el sistema y son beneficiados con él.

Tabla 2 Actor del sistema

Actor	Descripción
Editor Gráfico	Visualiza el proceso de simulación permitiendo realizar el diagrama de flujo de información (DFI) que luego se utiliza para lograr la simulación del proceso y obtener los resultados.

Casos de Uso del Sistema

Los casos de usos del sistema son considerados fragmentos de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores, es decir, son requerimientos funcionales que describen de una manera detallada el comportamiento del sistema con los distintos actores que interactúan con él. Son considerados operaciones importantes del sistema a construir.

En la descripción del sistema propuesto se detectaron lo siguientes casos de usos de sistema:

CU 1 Gestionar Proceso.

CU 2 Obtener propiedades de la corriente.

CU 3 Obtener propiedades de operación unitaria.

CU 4 Calcular operación unitaria.

CU 5 Evaluar propiedades.

CU 6 Identificar corriente.

CU 7 Identificar operación unitaria.

CU 8 Introducir corriente de entrada.

CU 9 Simular Proceso.

CU 10 Determinar Orden de Cálculo.

CU 11 Introducir propiedades de operación unitaria.

CU 12 Introducir propiedades de corrientes.

CU 13 Introducir Operación Unitaria.

CU 14 Manipular unidad de medida. Este caso de uso no se desarrollará en esta iteración de proyecto.

CU 15 Salvar Resultado.

Diagrama de Casos de Usos del Sistema

Un diagrama de casos de usos del sistema representa gráficamente a los procesos y su integración con los actores. Como una regla importante a tener en cuenta es que cada caso de uso debe comunicarse con

al menos un actor y si el caso de uso es abstracto (no instanciable) no tiene porque incluir relación con actores (aunque puede tenerlas). **(26)**

Patrones de Casos de Usos

Un patrón es definido como una pareja de problema/solución con un nombre que estandariza buenos principios y sugerencias. Existen diversas clases de patrones como de casos de usos, de arquitectura, de diseño entre otros. Los patrones de casos de usos permiten con precisión reflejar los requisitos reales, haciendo más fácil el trabajo con los sistemas y su mantenimiento. **(31)**

CRUD (Crear, Leer, Actualizar, Eliminar)

Este patrón se basa en la fusión de casos de usos simples para formar una unidad conceptual.

-Parcial

Este patrón alternativo modela una de las vías de los casos de usos como un caso de uso separado. Es preferiblemente utilizado cuando una de las alternativas de los casos de usos es más significativa, larga o más compleja que las otras. **(31)**

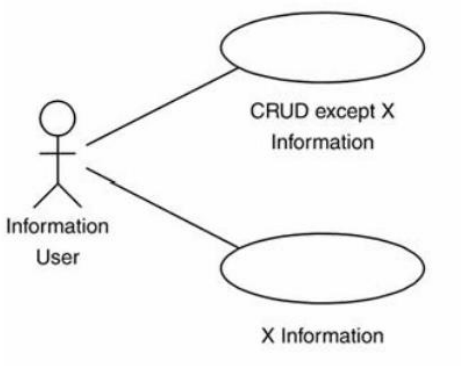

Ilustración 8 Patrón CRUD Parcial (31)

Extensión

El patrón de extensión consiste en dos casos de uso con una relación de extensión entre ellos. El caso de uso extendido es concreto, es decir, puede ser una instancia por su cuenta, así como ampliar la base de casos de uso. Estos últimos pueden ser concretos o abstractos. **(31)**

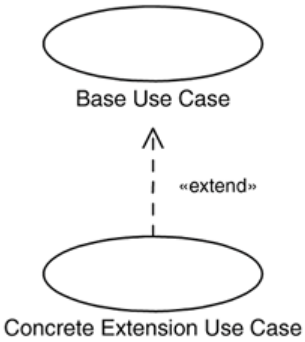

Ilustración 9 Patrón de Extensión (31)

Inclusión

En este patrón existe una relación de inclusión entre el caso de uso base a el caso de uso de inclusión. Esto último puede ser una instancia por su cuenta. El caso de uso base puede ser concreto o abstracto.

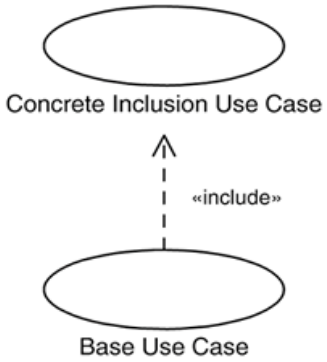

Ilustración 10 Patrón de Inclusión (31)

Concordancia

Extrae una subsecuencia de acciones que aparecen en diferentes lugares del flujo de casos de uso y es expresado por separado.

Reuso

Consta de 3 casos de uso. El primero llamado subsecuencia común, modela una secuencia de acciones que aparecerán en múltiples casos de uso en el modelo. Los otros casos de uso modelan el uso del sistema que comparte la subsecuencia común de acciones. De manera que deben existir al menos dos de ellos.

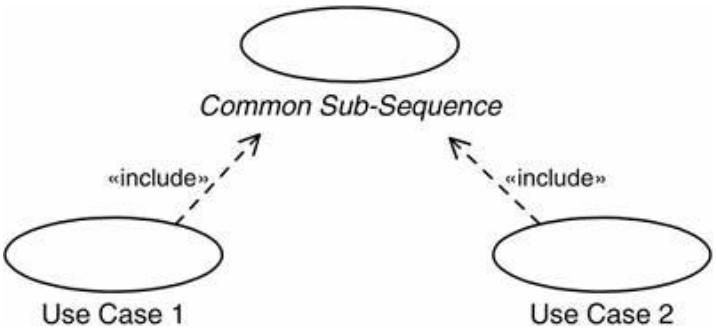

Ilustración 11 Patrón de Concordancia por Reuso (31)

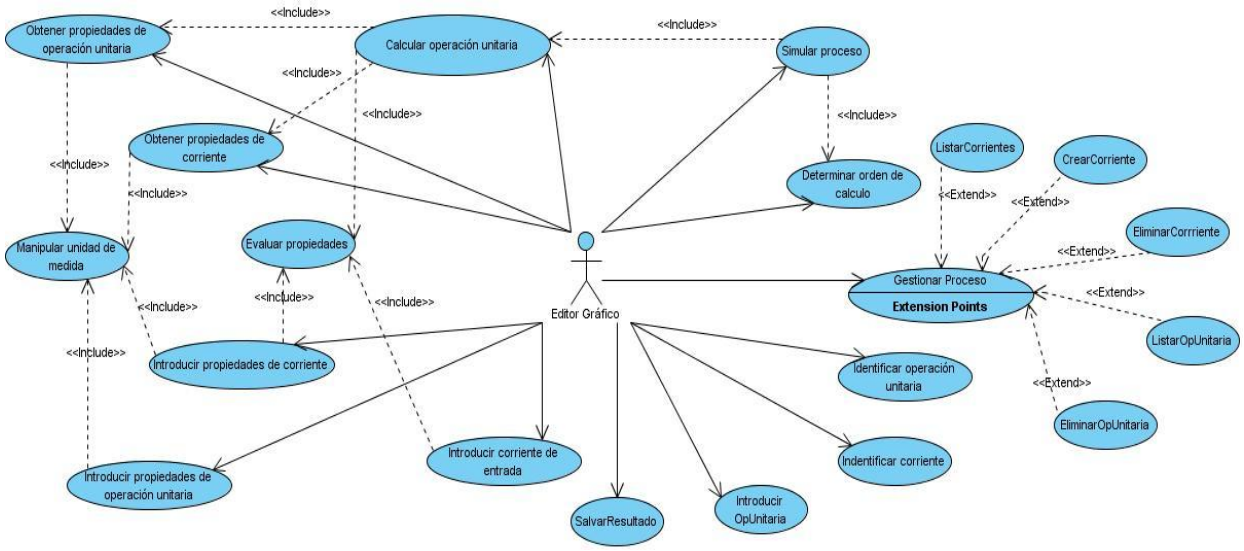

Ilustración 12 Diagrama de casos de usos del sistema

En el diagrama de casos de usos anteriormente expuesto se evidencian varios patrones como por ejemplo el CRUD parcial en el caso de uso Gestionar Proceso porque agrupa casos de usos como ListarCorriente, EliminarCorriente y AdicionarCorriente y no cuenta con ninguno que permita actualizar. El patrón de Concordancia por Reuso es aplicado en el caso de uso Manipular Unidad de Medida que es utilizado por Introducir Propiedad de Corriente e Introducir Propiedad de Operación Unitaria.

Descripción de los Casos de Usos del Sistema

A continuación se exponen una selección de las descripciones de los casos de usos mencionados anteriormente, por ser considerados críticos para el sistema. Para consultar las restantes descripciones ver Anexo1.

Tabla 3 CUS Gestionar Proceso

Caso de Uso:	Gestionar Proceso
Actores:	Editor Gráfico
Resumen:	El Caso de Uso se inicia cuando el Editor Gráfico invoca algunas de las funciones para gestionar proceso ya sea listar corriente, crear corriente, eliminar corriente, listar operaciones unitarias, adicionar operaciones unitarias o eliminar operación unitaria. El subsistema ejecuta la función invocada y termina el CU.
Precondiciones:	
Referencias	RF.1, RF1.1, RF1.2, RF1.3, RF1.4, RF1.5, RF1.6
Prioridad	Crítico

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Subsistema
1. El Editor Gráfico invoca alguna de las funciones para gestionar proceso.	2. El subsistema procesa la solicitud**.**
Sección “Listar corrientes”	
Acción del Actor	Respuesta del Subsistema
3. El Editor Gráfico invoca la función listar corrientes de un proceso.	4. El subsistema devuelve la lista de corrientes del proceso.
Sección “Crear corriente”	
Acción del Actor	Respuesta del Subsistema
5. El Editor Gráfico invoca la función crear nueva corriente e introduce los datos necesarios..	6. El subsistema comprueba los datos. 7. El subsistema crea la nueva corriente.
Sección “Eliminar corriente”	
Acción del Actor	Respuesta del Subsistema
8. El Editor Gráfico invoca la función eliminar corriente e introduce los datos necesarios.	9. El subsistema comprueba los datos. 10. El subsistema elimina la corriente.
Sección “Listar operaciones unitarias”	
Acción del Actor	Respuesta del Subsistema
11. El Editor Gráfico invoca la función listar operación unitaria.	12. El subsistema devuelve la lista de las unidades de operación.
Sección “Eliminar operación unitaria”	

Acción del Actor	Respuesta del Subsistema
15. El Editor Gráfico invoca la función eliminar operación unitaria del proceso e introduce los datos necesarios.	16. El subsistema valida los datos. 17. El subsistema elimina la operación unitaria del proceso.
Flujos Alternos	
Línea 6,9, 16: El subsistema lanza un mensaje de error porque los datos no son válidos.	
Poscondiciones	

Tabla 4 CUS Obtener Propiedades de Operación Unitaria

Caso de Uso:	Obtener propiedades de operación unitaria
Actores:	Editor Gráfico
Resumen:	El Caso de Uso se inicia cuando el Editor Gráfico invoca la función obtener propiedades de operación unitaria. El subsistema ejecuta la función invocada y termina el CU.
Precondiciones:	
Referencias	RF.3, CU incluido: Manipular Unidad de medida.
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Subsistema
1. El Editor Gráfico invoca la función obtener propiedades de operación unitaria e introduce los datos necesarios.	2. El subsistema comprueba los datos. 3. El subsistema devuelve las propiedades de la operación unitaria.
Flujos Alternos	
Línea 2: El subsistema lanza un mensaje de error porque los datos no son válidos.	
Poscondiciones	

Tabla 5 CUS Calcular Operación Unitaria

Caso de Uso:	Calcular operación unitaria.
Actores:	Editor Gráfico
Resumen:	El Caso de Uso se inicia cuando el Editor Gráfico invoca la función calcular operación unitaria. El subsistema ejecuta la función invocada y termina el CU.
Precondiciones:	Introducir los datos requeridos.

Referencias	RF.4, RF4.1, RF4.2, RF4.3 CU incluidos: Obtener propiedades de corriente, Obtener propiedades de operación unitaria, Evaluar propiedad
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Subsistema
1. El Editor Gráfico invoca la función calcular operación unitaria e introduce los datos necesarios.	2. El subsistema valida los datos. 3. El subsistema obtiene los componentes y propiedades de las corrientes de entrada, así como las propiedades de la operación unitaria asociada a la misma. 4. El subsistema calcula los componentes y propiedades de las corrientes de salida y los parámetros adicionales de la operación unitaria. 5. El subsistema actualiza las propiedades de las operaciones unitarias y de las corrientes del proceso sus componentes y propiedades. 6. Ejecuta CU Evaluar propiedad.
Flujos Alternos	
Línea 2: El subsistema lanza un mensaje de error porque los datos no son válidos.	
Poscondiciones	

Tabla 6 CUS Identificar Operación Unitaria

Caso de Uso:	Identificar operación unitaria.
Actores:	Editor Gráfico
Resumen:	Esta funcionalidad permite identificar operaciones unitarias del proceso.
Precondiciones:	
Referencias	RF6
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Subsistema
1. El Editor Gráfico invoca la función identificar operación unitaria e introduce los datos necesarios.	2. El subsistema valida los datos. 3. El subsistema actualiza los datos (nombre e identificador).

Flujos Alternos	
Línea 2: El subsistema lanza un mensaje de error porque los datos no son válidos.	
Poscondiciones	

Tabla 7 CUS Introducir Corriente de Entrada

Caso de Uso:	Introducir Corriente de Entrada
Actores:	Editor Gráfico
Resumen:	El Caso de Uso se inicia cuando el Editor Gráfico invoca la función introducir corriente de entrada. El subsistema ejecuta y termina el CU.
Precondiciones:	
Referencias	RF.8, RF 8.1, CU incluidos: Evaluar Propiedades
Prioridad	Critico
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Subsistema
1. El subsistema Editor Gráfico invoca la función introducir corriente de entrada e introduce los datos necesarios.	2. El subsistema valida los datos. 3. El subsistema crea la nueva corriente con dichos valores.
Flujos Alternos	
Línea 2: El subsistema lanza un mensaje de error porque los datos no son válidos.	
Poscondiciones	

Tabla 8 CUS Simular Proceso

Caso de Uso:	Simular Proceso
Actores:	Editor Gráfico
Resumen:	El Caso de Uso se inicia cuando el Editor Gráfico invoca la función simular proceso. El subsistema ejecuta y termina el CU.
Precondiciones:	Tener los datos requeridos.
Referencias	RF.9, CU incluidos: Determinar Orden de Cálculo, Calcular operación unitaria.
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Subsistema

1. El Editor Gráfico invoca la función simular proceso.	2. El subsistema invoca al CU Determinar Orden de Cálculo. 3. El subsistema invoca al CU Calcular operación unitaria para cada una de las operaciones unitarias presentes en el proceso.
Flujos Alternos	
Línea 4: El subsistema lanza un mensaje de error porque no se pueden realizar la simulación y los motivos.	
Poscondiciones	Genera resultados para posterior análisis.

Tabla 9 CUS Introducir Propiedades de Operación Unitaria

Caso de Uso:	Introducir propiedades de operación unitaria
Actores:	Editor Gráfico
Resumen:	El Caso de Uso se inicia cuando el Editor Gráfico invoca la función introducir propiedades de operación unitaria. El subsistema ejecuta y termina el CU.
Precondiciones:	
Referencias	R.F11 CU incluido: Manipular Unidad de medida
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Subsistema
1. El Editor Gráfico invoca la función introducir propiedades de operación unitaria e introduce los datos necesarios.	2. El subsistema valida los datos. 3. El subsistema actualiza las propiedades de la operación unitaria.
Flujos Alternos	
Línea 2: El subsistema lanza un mensaje de error porque los datos no están correctos.	
Poscondiciones	

Conclusiones

Después de haber realizado parte del modelado de la solución se tiene una visión clara y precisa del dominio del problema que aborda la investigación. Se recopilaron varios requerimientos no funcionales teniendo en cuenta las cualidades que se deben cumplir para poseer una maquina virtual de java. También se recogieron trece requerimientos funcionales algunos de los cuales englobaban otros. Estos fueron agrupados en quince casos de usos y un actor del sistema en un diagrama de casos de usos del sistema, parte fundamental de este capítulo.

CAPÍTULO 3: Construcción de la solución propuesta.

Introducción

En este capítulo se aborda la última parte del flujo de trabajo de modelado, implementación completo y para detectar defectos en el código se realizan pruebas unitarias. Inicialmente se justifica la opción tomada de no utilizar el análisis. Luego aparece el diseño de la solución exponiéndose el diagrama de clases del diseño con los respectivos diagramas de secuencia de cada uno de los casos de usos existentes.

Análisis

El análisis tiene como objetivo conseguir una compresión más precisa de los requisitos con una descripción más fácil de mantener y que ayuda a estructurar el sistema entero pero con la gran diferencia de que puede utilizarse el lenguaje de los desarrolladores para describir los resultados.
El modelo de análisis es considerado una primera aproximación y entrada fundamental al modelo del diseño esto es porque debe ser mantenido el sistema en su conjunto y no solo los requisitos.

¿Por qué no utilizar el análisis?

El análisis sin duda ejerce un papel importante en el ciclo de vida del software pero su empleo difiere de un proyecto a otro. A continuación se enuncian ejemplos de situaciones que pueden existir con la aplicación del análisis:

1- El proyecto utiliza el modelo del análisis para describir los resultados del análisis y mantiene la consistencia de este modelo a lo largo de todo el ciclo de vida del software.
2- El proyecto utiliza el modelo del análisis para describir los resultados del análisis pero considera a este modelo como una herramienta transitoria e intermedia. Más adelante cuando el diseño y la implementación están en marcha se deja de actualizar el análisis. En su lugar cualquier tema de análisis que aún quede se resuelve como parte integrada dentro del trabajo del diseño en el modelo del diseño resultante.
3- El proyecto no utiliza en absoluto el modelo del análisis para describir los resultados del análisis. En cambio el proyecto analiza los requisitos como parte integrada en la captura de requerimientos o en el diseño. (26)

La última variante es la aplicada en esta investigación porque los requisitos son simples y para evitar el coste en tiempo de mantener el modelo del análisis. También la metodología AUP que se está utilizando es bastante flexible solicitando solo entregables mínimos.

Diseño

En el diseño se modela el sistema para que soporte todos los requisitos siendo su principal resultado el modelo del diseño.

El modelo del diseño es un modelo de objeto que describe la realización física de los casos de usos centrándose en como los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación tienen impacto en el sistema.

Diagrama de Clases del Diseño

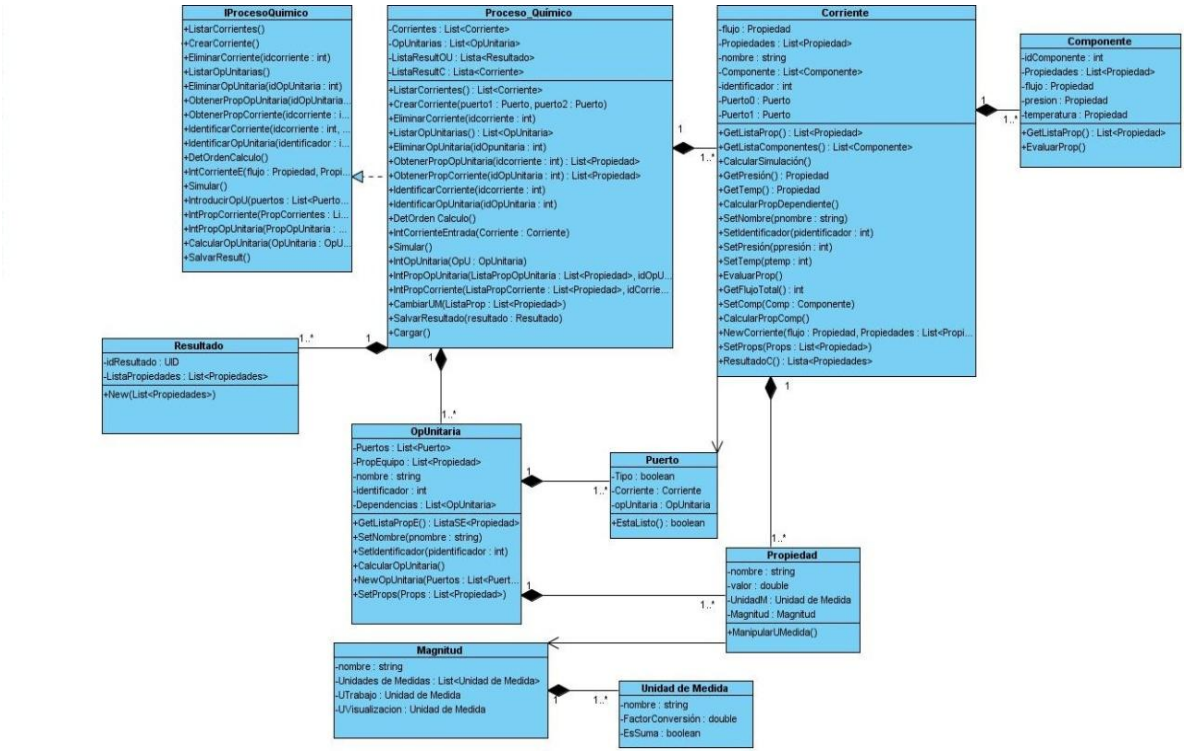

Ilustración 13 Diagrama de clases del diseño

Patrones de Diseño

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular. Su principal objetivo es el de identificar Clases, Instancias, Roles, Colaboraciones y la distribución de responsabilidades.

Patrones GOF (Gang-Of-Four)

Cada patrón describe un problema que ocurre una y otra vez en nuestro ambiente, y luego describe el núcleo de la solución a ese problema, de tal manera que puedes usar esa solución un millón de veces más, sin hacer jamás la misma cosa dos veces.

Existen 3 tipos de patrones GOF:

- De Creación: Abstraen el proceso de creación de instancias.
- Estructurales: Se ocupan de cómo clases y objetos son utilizados para componer estructuras de mayor tamaño.
- De Comportamiento: Atañen a los algoritmos y a la asignación de responsabilidades entre objetos.

Dentro de los patrones estructurales se utiliza el Fachada (Facade) en la interfaz IProceso_Químico ya que constituye una interfaz que facilita al cliente el uso del subsistema, así como proporciona un acoplamiento débil entre el subsistema y el cliente.

Patrones para asignar responsabilidades a utilizar (GRASP)

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. A continuación se describen los patrones de asignación de responsabilidades empleados para la obtención de un diseño eficaz.

Patrón Experto:

Problema:

¿Cuál es el principio fundamental en virtud del cual se asignan las responsabilidades en el diseño orientado a objetos?

Solución:

Asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad.

Patrón Creador:

Problema:

¿Quién debería ser responsable de crear una nueva instancia de alguna clase?

Solución:

Asignarle a la clase B la responsabilidad de crear una instancia de clase A en uno de los siguientes casos:

- B agrega los objetos A.
- B contiene los objetos A.

- B registra las instancias de los objetos A
- B utiliza especialmente los objetos A.
- B tiene los datos de inicialización que serán transmitidos a A cuando este objeto sea creado (así que B es un Experto respecto a la creación de A). B es un creador de los objetos A.

Si existe más de una opción, prefiera la clase B que agregue o contenga la clase A.

Patrón Bajo Acoplamiento:

Problema:

¿Cómo dar soporte a una dependencia escasa y a un aumento de la reutilización?
El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas.

Solución:

Asignar una responsabilidad para mantener bajo acoplamiento.

El grado de acoplamiento no puede considerarse aisladamente de otros principios como Experto y Alta Cohesión.

Patrón Alta Cohesión:

Problema:

¿Cómo mantener la complejidad dentro de límites manejables?

La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase.

Solución:

Asignar una responsabilidad de modo que la cohesión siga siendo alta.

Para el diseño de las clases tuvimos en cuenta el patrón Experto por ejemplo clases como Corriente y OpUnitaria son las encargadas de gestionar su información ya que estas poseen lo necesario para cumplir con su responsabilidad. También se tuvo en cuenta el patrón de alta cohesión en la clase Componente al darle la funcionalidad de evaluar sus propias propiedades. Se aplicó el patrón Creador en la clase ProcesoQuímico ya que contiene muchos objetos Corrientes convirtiéndose en la idónea para crear instancias de estas últimas. El patrón de Bajo Acoplamiento se empleó para evitar el uso de relaciones innecesarias entre las clases.

Diagramas de Interacción del diseño

El diagrama de interacción muestra la realización de un flujo o escenario concreto de un caso de uso en términos de interacción entre objetos del diseño. De este diagrama existen dos tipos, el de colaboración y el de secuencia.

En el diseño es preferible representar diagramas de secuencia ya que el centro de atención principal es el de encontrar secuencias de interacciones detalladas y ordenadas en el tiempo.

Diagramas de Secuencia

El diagrama de secuencia muestra la interacción entre objetos mediante transferencia de mensajes entre objetos o subsistemas. Si los casos de usos tienen varios flujos o subflujos distintos se hace necesario crear uno para cada uno de ellos. Posteriormente se ilustran una selección de los diagramas de secuencia de cada uno de los casos de usos del sistema, para ver los diagramas restantes consultar Anexo2.

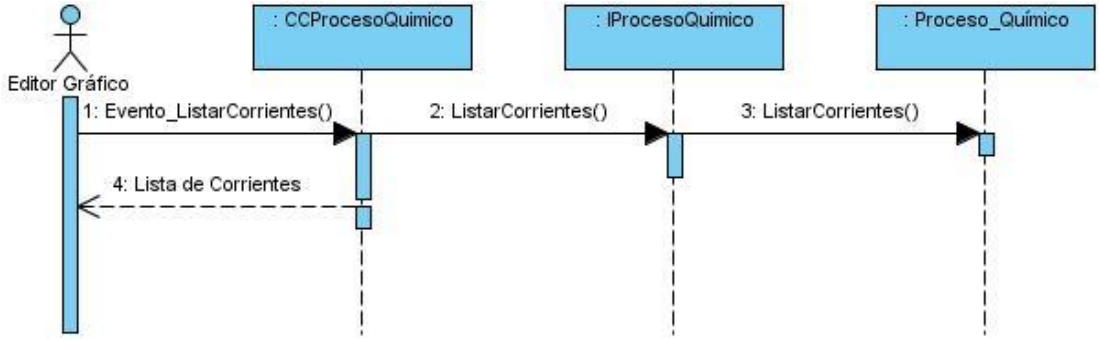

Ilustración 14 DS Gestionar Proceso (Listar Corriente)

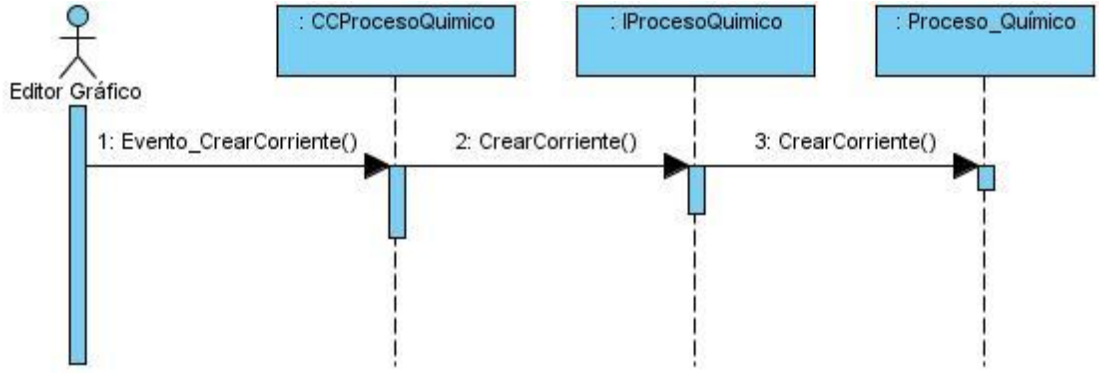

Ilustración 15 DS Gestionar Proceso (Crear Corriente)

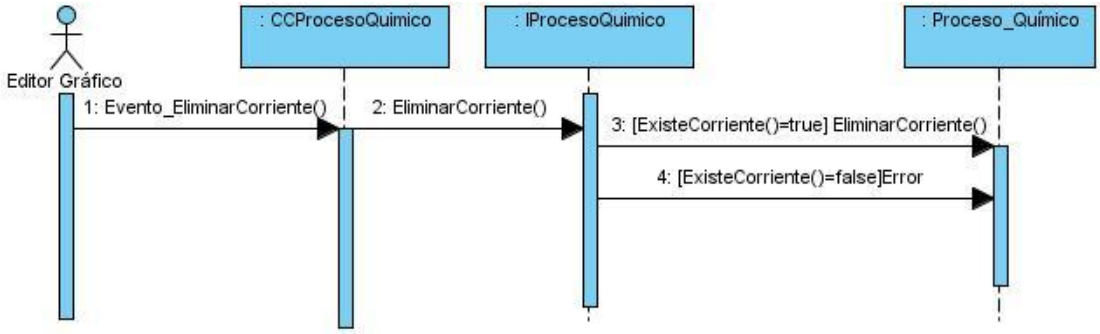

Ilustración 16 DS Gestionar Proceso (Eliminar Corriente)

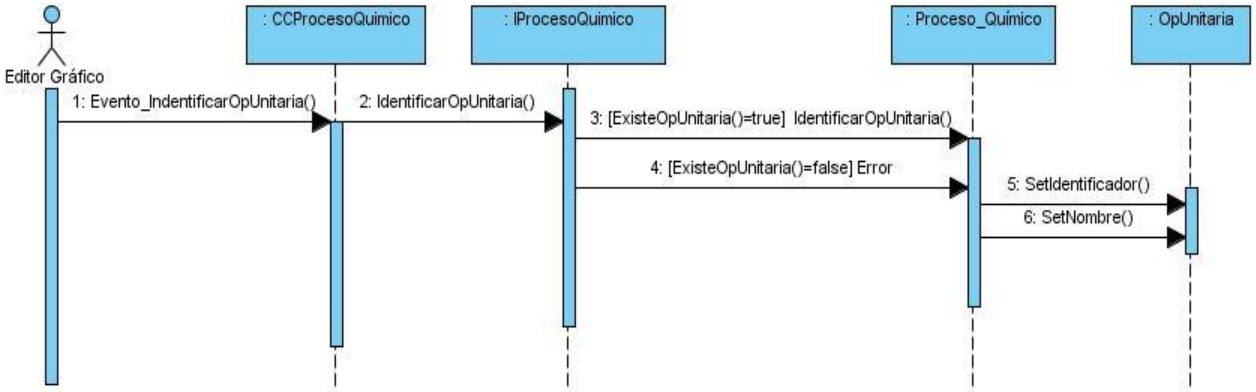

Ilustración 17 DS Identificar Operación Unitaria

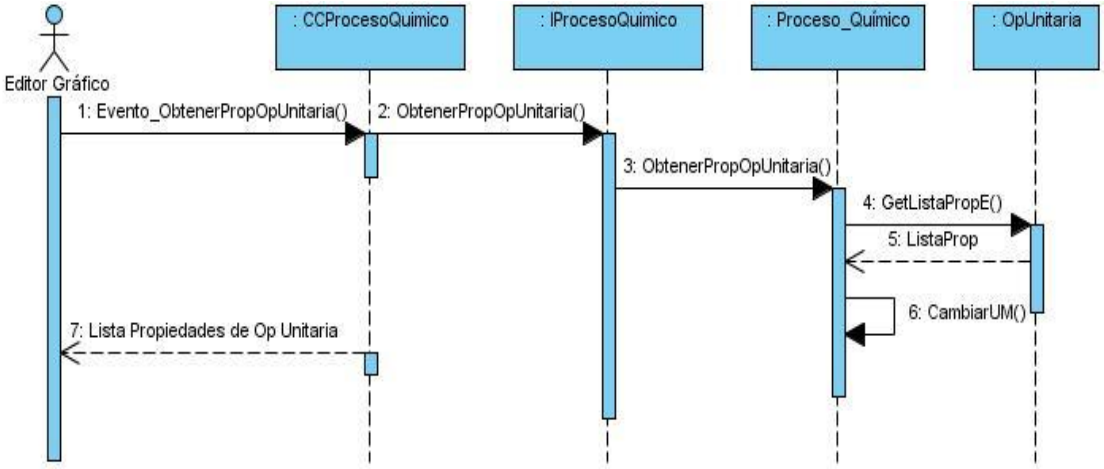

Ilustración 18 DS Obtener Propiedades de Operación Unitaria

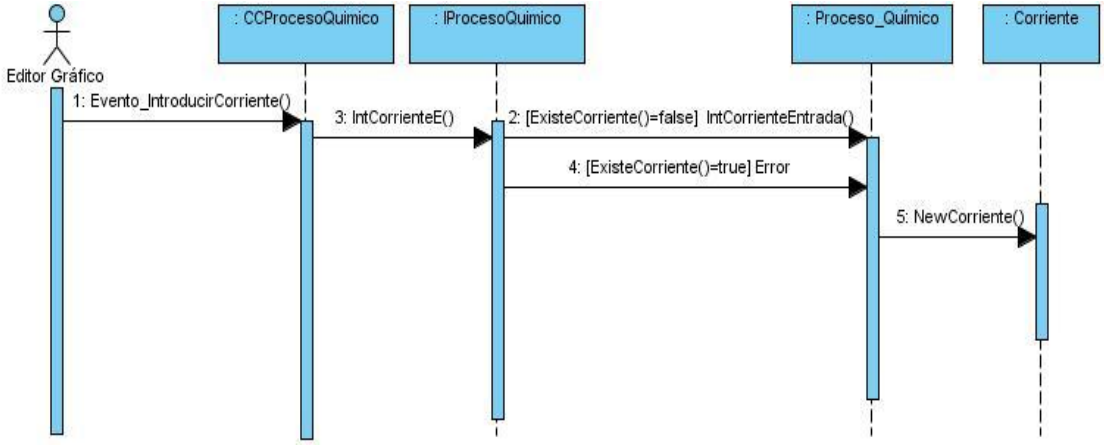

Ilustración 19 DS Introducir Corriente de Entrada

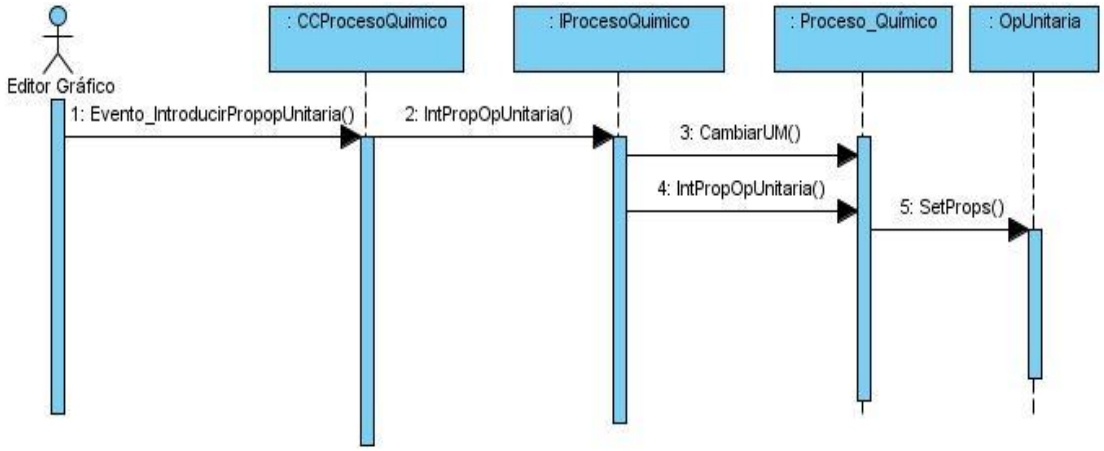

Ilustración 20 DS Introducir Propiedades de Operación Unitaria

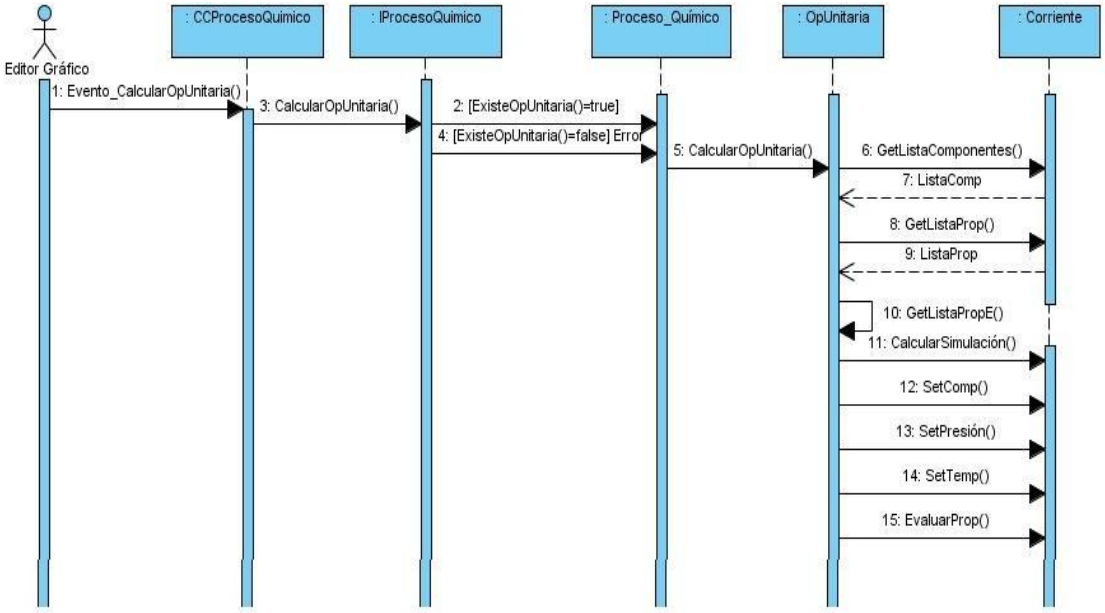

Ilustración 21 DS Calcular Operación Unitaria

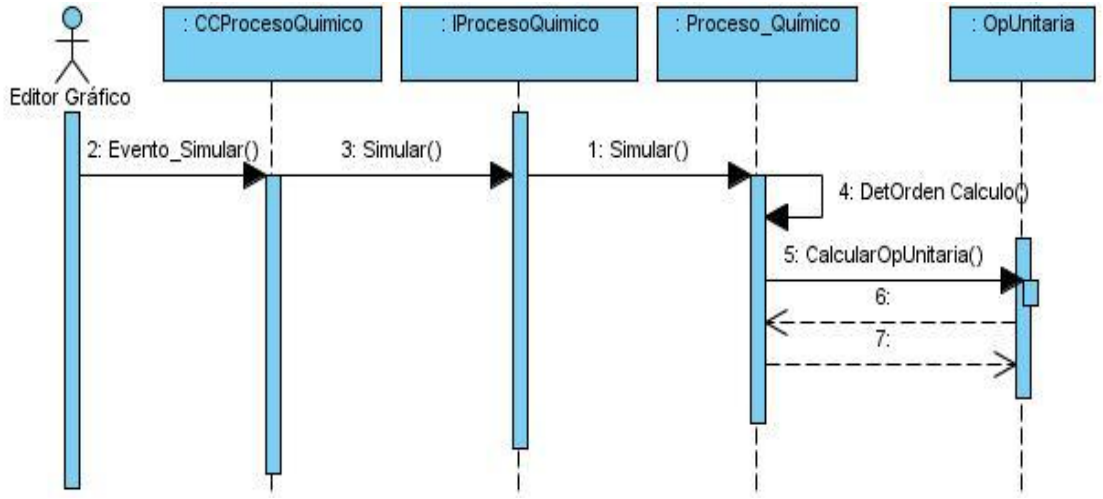

Ilustración 22 DS Simular Proceso

Patrón de Arquitectura a utilizar

El Modelo-Vista-Controlador (MVC) es el patrón de arquitectura de software que se va a utilizar en el desarrollo de la solución. El MVC fue diseñado para reducir el esfuerzo al momento de programar. Además porque permite una clara separación entre los componentes de un programa; implementándolos por separado y posibilitando al equipo trabajar con componentes más eficientes.

Descripción del patrón

• Modelo: Esta es la representación específica de la información con la cual el sistema opera. La lógica de datos asegura la integridad de estos y permite derivar nuevos datos. De las tres entidades que presenta, el modelo es el corresponde implementar en esta investigación.

• Vista: Este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.

• Controlador: Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

Su característica principal es que cualquier cambio producido en el Modelo se refleje automáticamente en cada una de las Vistas.

¿Por qué MVC?

El patrón de diseño MVC se emplea principalmente en sistemas de representación gráfica de datos porque cumple con uno de los enunciados más antiguos de la programación distribuida, separar la presentación del acceso a datos y la lógica de negocios. Java representa uno de los pilares más

importantes en desarrollo de aplicaciones distribuidas, ya que todos sus programas tienen un marco que contiene todos los elementos, un controlador de eventos, cálculos y la presentación del resultado. Es por eso que el MVC está orientado al lenguaje más abierto y portable que se haya construido.

Implementación

En la implementación se comienza con el resultado del diseño e implementamos el sistema en términos de componentes, es decir, ficheros de código fuente, scripts, ficheros de código binario, ejecutables y similares.

El modelo de implementación describe como se organizan los componentes de acuerdo con los mecanismos de estructuración y modulación disponibles en el entorno de implementación y en el lenguaje utilizado y como dependen los componentes unos con otros.

Diagrama de Componentes

Un diagrama de componentes muestra las dependencias lógicas entre componentes de software, sean éstos componentes fuentes, binarios o ejecutables. Un componente representa una unidad de código (fuente, binario o ejecutable) que permite mostrar las dependencias en tiempo de compilación y ejecución.

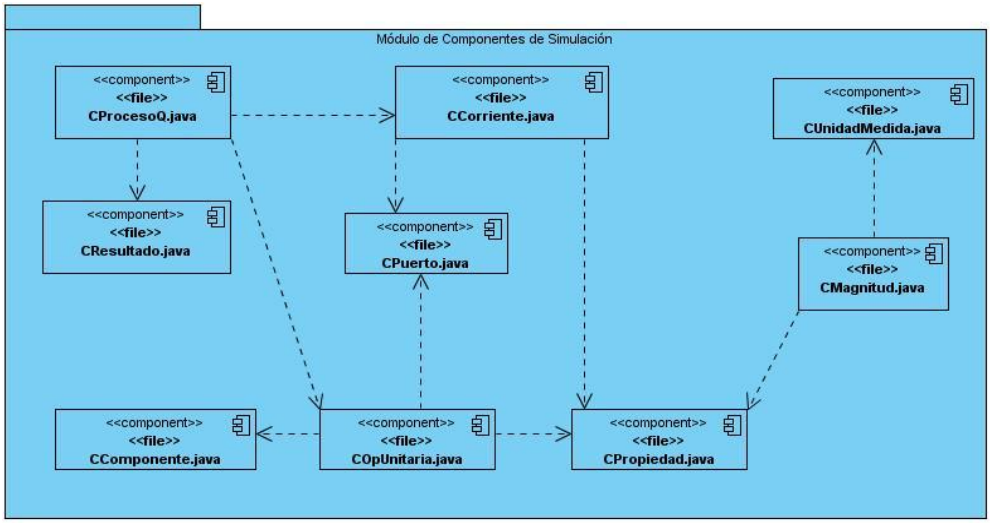

Ilustración 23 Diagrama de Componentes

Diagrama de Despliegue

Un diagrama de despliegue muestra la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes software (procesos y objetos que se ejecutan en ellos). Es un grafo de nodos unidos por conexiones de comunicación donde un nodo puede contener instancias de componentes de software, objetos y procesos (caso particular de un objeto).

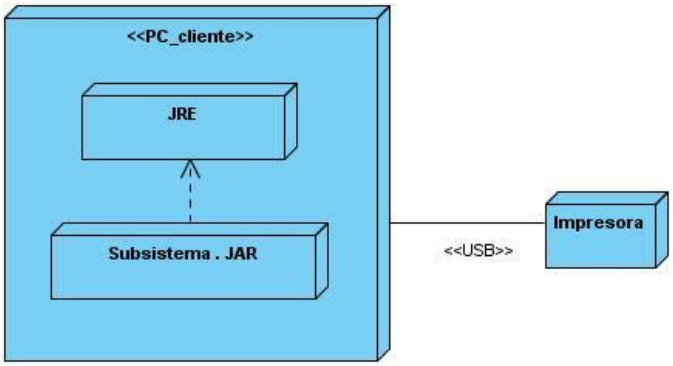

Ilustración 24 Diagrama de Despliegue

Prueba del sistema propuesto

Las pruebas es una actividad en la cual un sistema o componente es ejecutado bajo unas condiciones o requerimientos específicos, los resultados son observados y registrados realizando una evaluación de algún aspecto del sistema o componente.

En los últimos años se han desarrollado un conjunto de herramientas denominadas XUnit que facilitan la elaboración de pruebas en diferentes lenguajes. JUnit forma parte de dicho conjunto siendo la herramienta para realizar pruebas de integración y de unidades en Java.

Para realizar las pruebas de esta investigación se utilizó JUnit por ser un marco de trabajo de código abierto y libre para la automatización de las pruebas de unidades o de caja blanca, facilitando así la corrección de métodos y objetos.

La prueba de unidad prueba el menor de los componentes de un sistema de manera aislada. Cada una de esas unidades define un conjunto de estímulos (llamada de métodos), y de datos de entrada y salida asociados a cada estímulo. Las entradas son parámetros y las salidas son el valor de retorno, excepciones o el estado del objeto.

Algunas ventajas de utilizar JUnit:

1. Permite la creación rápida de código de prueba mientras posibilita un aumento en la calidad del sistema siendo desarrollado y probado.
2. Ampliamente utilizado por los desarrolladores de la comunidad código-abierto, existiendo un gran número de ejemplos.
3. Una vez escritos, las pruebas son ejecutadas rápidamente sin que, para eso, sea interrumpido el proceso de desarrollo.
4. JUnit chequea los resultados de las pruebas y suministra una respuesta inmediata.
5. Se puede crear una jerarquía de pruebas que permitirá probar sólo una parte del sistema o todo él.

6. Escribir pruebas con JUnit permite que el desarrollador pierda menos tiempo depurando su código.

El concepto fundamental en estas herramientas es el caso de prueba (test case), y la suite de prueba (test suite). Los casos de prueba son clases o módulos que disponen de métodos para probar las funcionalidades de un clase o módulo concreta/o. Así, para cada clase que quisiéramos probar definiríamos su correspondiente clase de caso de prueba. Mediante las suites podemos organizar los casos de prueba, de forma que cada suite agrupa los casos de prueba de módulos que están funcionalmente relacionados. Típicamente una prueba unitaria ejecuta un método individualmente y compara una salida conocida después del procesamiento de la misma. Por ejemplo:

Assert.assertEquals (2, Método(1));

La expresión anterior verifica si la salida de Método () es 2, cuando ese método recibe el parámetro 1.

A cada funcionalidad del subsistema se le aplicaron pruebas con el Junit las que demostraron en algunas ocasiones la existencia de errores en el código. Un ejemplo fue en el método Determinar Orden de Cálculo, que luego de su primera ejecución al verificar, se había quedado la lista de dependencia vacía. También fueron probadas funcionalidades como Buscar Corriente, Identificar Corriente y Operación Unitaria.

Beneficios del formato de ficheros JAR

El subsistema desarrollado es un formato de ficheros JAR (Archivos Java) o biblioteca de clases que gestiona procesos químicos compuestos por operaciones unitarias, corrientes y los resultados de sus simulaciones. Un fichero JAR proporciona beneficios como:

- Seguridad: Puedes firmar digitalmente el contenido de un fichero JAR. Los usuarios que reconozcan tu firma pueden permitir a tu software privilegios de seguridad que de otro modo no tendría.
- Compresión: El formato JAR permite comprimir los ficheros para ahorrar espacio.
- Empaquetado versionado: Un fichero JAR puede contener datos sobre los ficheros que contiene, como información sobre la versión.
- Portabilidad: El mecanismo para manejar los ficheros JAR son una parte estándar del corazón del API de la plataforma Java. Un JAR se puede ejecutar en Windows, Linux, Macintosh o en cualquier sistema operativo solo se necesita la maquina virtual de java.

Conclusiones

En la construcción de la solución viable se realizó un diagrama de clases del diseño y diagramas de secuencia para cada uno de los casos de usos del sistema. Como la interacción entre los objetos es muy variable fue necesario detallar todas las posibles secuencias para el siguiente flujo de trabajo de implementación. En la etapa de implementación a partir de los resultados obtenidos anteriormente se realizó un diagrama de componentes representado como un grafo de componentes de software unidos por medio de relaciones de dependencia (compilación, ejecución). También se confeccionó un diagrama de despliegue para mostrar las relaciones físicas entre los componentes de hardware y software en el sistema final. Como parte del flujo de trabajo prueba se desarrollaron las pruebas de unidad utilizando JUnit versión 4.8.2 como marco de trabajo lográndose un aumento de la confianza en el código, en la misma proporción que el volumen de pruebas que se realizó.

Conclusiones Generales

A lo largo de este trabajo se han definido una serie de elementos importantes que forman parte de la simulación de procesos químicos, iniciando por la fundamentación teórica hasta la implementación en la computadora, todo con el objetivo de cubrir la necesidad de crear una herramienta que permita gestionar los componentes químicos, propiedades físico-químicas y modelos de operaciones unitarias.

Se inició con la investigación de lo que es simulación de procesos, sus ventajas y desventajas, las clasificaciones que posee, así como se caracterizaron principales simuladores de procesos tanto internacionales como nacionales.

En la selección de la metodología y las herramientas se adoptó el paradigma de software libre ya que facilita en gran medida la reutilización de código y un desarrollo acelerado de aplicaciones, posibilitando el estudio exhaustivo de código existente afín con la aplicación que se diseña.

En la construcción del subsistema se transito por diferentes flujos de trabajos como Modelado, Implementación y Prueba utilizando técnicas ágiles con la mínima documentación.

El producto final de toda la investigación aunque no cuenta con todos los elementos de una simulación comercial constituye un paso de avance en el diseño de nuevos y mejores simuladores de procesos químicos cubanos.

Recomendaciones

Se recomiendan los siguientes aspectos al trabajo:

- Continuar el desarrollo de nuevas funcionalidades que amplíen las prestaciones del subsistema, por ejemplo la conversión de unidades de medida.
- Integrar el subsistema obtenido con el subsistema Editor Gráfico y Análisis de los Resultados.
- Optimizar la aplicación teniendo en cuenta la implementación de tipos de datos abstractos.

Referencias bibliográficas

1. Sifuentes, Victor Hugo Martínez, y otros. ***Simulación de Procesos en Ingeniería Química.*** **México : s.n., 2000.**

2. Corrales Valdés, Yurisnel y Suarez López, Juan Carlos. ***Desarrollo de la arquitectura del sistema para la Simulación de Procesos en la Industria Química Cubana.*** **Universidad de Ciencias Informáticas. Ciudad de la Habana : s.n., 2007.**

3. Dale L, Bolton y Donald P, Anderson. ***El empleo de la simulación en la administración educacional.*** **México : Paidos, 1971.**

4. R. F., Shannon. ***Simulation: A Survey with Research Suggestions.*** **1975.**

5. Scenna, Dr Nicolás J. **Modelado, Simulación y Optimización de Procesos Químicos. [En línea] 1999. http:// www.modeladpeningenieria.edu.ar/libro/cap05.pdf.**

6. Pro Bermejo, Rafael. ***Modelos de simulación .*** **España : s.n., 1978.**

7. Hoeger, Prof Herbert. ***Simulación.*** **2009.**

8. Texson Alarcón, Griselda Guadalupe. ***Diseño de un simulador de vuelo para la compra y venta de acciones en el mercado accionario mexicano.*** **Universidad de Américas Puebla. México : s.n., 2005.**

9. Costa López, José, y otros. ***Curso de Ingeniería Química.*** **2002.**

10. Pérez Ones, Ing Osney y Gozá León, Dr Ing Osvaldo. ***Simulación mudular secuencial de destilerías de alcohol con fines energéticos.*** **CUJAE. Ciudad de la Habana : s.n., 2004.**

11. E Tarifa, Enrigue. **Simulación de sistemas químicos. [En línea] 1999. http://www.modeladoeningenieria.edu.ar/libro.**

12. Banks, J, Carson, J S y B J, Nelson. ***Discrete- Event System Simulation.*** **New Jersey : s.n., 1996.**

13. Espinosa Sosa, IA. ***Minimización de la entropía generada y la energía destruida en los cicloa termodinámicos.*** **2003.**

14. Hyprotech. ***HYSYS. Process Documentación Suite. Calgary: Hyprotech.*** **1998.**

15. Henao, C y Vélez, J. ***Manual del laboratorio diseño de procesos químicos - Uso del paquete de simulación HYSYS. Process. UPB.*** **Medellín : s.n., 2002.**

16. Herrera Orozco, Isael. ***Desarrollo metodológico de evaluación ambiental en el análisis de procesos.*** **2004.**

17. Velázquez Batista, Adalberto. ***Informe del rol de analista del subsistema de modelos y propiedades de un simulador para la industria química.*** **2009.**

18. Brown, Steve. ***CHEMCAD Integration.*** **Texas : s.n., 2007.**

19. —. ***ConcepSys CHEMCAD Integration.*** **Texas : s.n., 2007.**

20. Suárez López, Juan Carlos y Corrales Valdés, Yurisnel. ***Simulador de la industria química. Documento de descripción de la arquitectura.*** **2007.**

21. González García, Moisés. ***Método de Desarrollo Arquitectónico en Grupo.*** **México : s.n., 2006.**

22. Passerini, Ing Nicolás y A. Brey, Ing Gustavo. ***Metodologías Iterativas de Desarrollo.*** **Dpto de Ciencias, Universidad Tecnológica Nacional. Buenos Aires : s.n., 2005.**

23. W. Ambler, Scott. ***Software Development.Unified and Agile .*** **2006.**

24. Manzanero del Campo, Miguel Angel y Cruzado Nuño, Ignacio. ***Guía de iniciación al lenguaje Java.*** **1999.**

25. Bermejo Sanz, Laura y Gómez Monreal, Enrique. ***Eclipse como IDE.*** **2007.**

26. Jacobson, Ivar, Booch, Grady y Rumbaugh, James. ***El proceso unificado de desarrollo de software.*** **La Habana : s.n., 2004.**

27. Engels, y otros. ***UML-A Universal Languaje?*** **Berlin : s.n., 2000.**

28. Sommerville, Ian. ***Ingeniería del Software.*** **España : s.n., 2006.**

29. Pupo Polanco, Rosell y González Pérez, Yenier. ***Implementación del componente réplica de base de datos para Akademos v2.0.*** **Universidad de las Ciencias Informáticas. Ciudad de la Habana : s.n., 2009.**

30. Pérez Lovelle, Sonia. ***Automatización de la Arquitectura de Componentes Genéricos usando UML.*** **CUJAE. Ciudad de la Habana : s.n., 2005.**

Bibliografía

1. Sifuentes, Victor Hugo Martínez, et al. ***Simulación de Procesos en Ingeniería Química.*** **México : s.n., 2000.**

2. Corrales Valdés, Yurisnel and Suarez López, Juan Carlos. ***Desarrollo de la arquitectura del sistema para la Simulación de Procesos en la Industria Química Cubana.*** **Universidad de Ciencias Informáticas. Ciudad de la Habana : s.n., 2007.**

3. Dale L, Bolton and Donald P, Anderson. ***El empleo de la simulación en la administración educacional.*** **México : Paidos, 1971.**

4. R. F., Shannon. ***Simulation: A Survey with Research Suggestions.*** **1975.**

5. Scenna, Dr Nicolás J. **Modelado, Simulación y Optimización de Procesos Químicos. [Online] 1999. http:// www.modeladpeningenieria.edu.ar/libro/cap05.pdf.**

6. Pro Bermejo, Rafael. ***Modelos de simulación .*** **España : s.n., 1978.**

7. Hoeger, Prof Herbert. ***Simulación.*** **2009.**

8. Texson Alarcón, Griselda Guadalupe. ***Diseño de un simulador de vuelo para la compra y venta de acciones en el mercado accionario mexicano.*** **Universidad de Américas Puebla. México : s.n., 2005.**

9. Costa López, José, et al. ***Curso de Ingeniería Química.*** **2002.**

10. Pérez Ones, Ing Osney and Gozá León, Dr Ing Osvaldo. ***Simulación mudular secuencial de destilerías de alcohol con fines energéticos.*** **CUJAE. Ciudad de la Habana : s.n., 2004.**

11. E Tarifa, Enrigue. **Simulación de sistemas químicos. [Online] 1999. http://www.modeladoeningenieria.edu.ar/libro.**

12. Banks, J, Carson, J S and B J, Nelson. ***Discrete- Event System Simulation.*** **New Jersey : s.n., 1996.**

13. Espinosa Sosa, IA. ***Minimización de la entropía generada y la energía destruida en los cicloa termodinámicos.*** **2003.**

14. Hyprotech. ***HYSYS. Process Documentación Suite. Calgary: Hyprotech.*** **1998.**

15. Henao, C and Vélez, J. ***Manual del laboratorio diseño de procesos químicos - Uso del paquete de simulación HYSYS. Process. UPB.*** **Medellín : s.n., 2002.**

16. Herrera Orozco, Isael. ***Desarrollo metodológico de evaluación ambiental en el análisis de procesos.*** **2004.**

17. Velázquez Batista, Adalberto. ***Informe del rol de analista del subsistema de modelos y propiedades de un simulador para la industria química.*** **2009.**

18. Brown, Steve. ***CHEMCAD Integration.*** **Texas : s.n., 2007.**

19. —. ***ConcepSys CHEMCAD Integration.*** **Texas : s.n., 2007.**

20. Suárez López, Juan Carlos and Corrales Valdés, Yurisnel. ***Simulador de la industria química. Documento de descripción de la arquitectura.*** **2007.**

21. González García, Moisés. ***Método de Desarrollo Arquitectónico en Grupo.*** **México : s.n., 2006.**

22. Passerini, Ing Nicolás and A. Brey, Ing Gustavo. ***Metodologías Iterativas de Desarrollo.*** **Dpto de Ciencias, Universidad Tecnológica Nacional. Buenos Aires : s.n., 2005.**

23. W. Ambler, Scott. ***Software Development.Unified and Agile .*** **2006.**

24. Manzanero del Campo, Miguel Angel and Cruzado Nuño, Ignacio. ***Guía de iniciación al lenguaje Java.*** **1999.**

25. Bermejo Sanz, Laura and Gómez Monreal, Enrique. ***Eclipse como IDE.*** **2007.**

26. Jacobson, Ivar, Booch, Grady and Rumbaugh, James. ***El proceso unificado de desarrollo de software.*** **La Habana : s.n., 2004.**

27. Engels, et al. ***UML-A Universal Languaje?*** **Berlin : s.n., 2000.**

28. Sommerville, Ian. ***Ingeniería del Software.*** **España : s.n., 2006.**

29. Pupo Polanco, Rosell and González Pérez, Yenier. ***Implementación del componente réplica de base de datos para Akademos v2.0.*** **Universidad de las Ciencias Informáticas. Ciudad de la Habana : s.n., 2009.**

30. Pérez Lovelle, Sonia. ***Automatización de la Arquitectura de Componentes Genéricos usando UML.*** **CUJAE. Ciudad de la Habana : s.n., 2005.**

31. Pérez de Alejo, Hector Eugenio and Hernández León, Rolando Alfredo. ***Informatización de la ingeniería de proceso en la industria química cubana.*** **Ciudad de la Habana : s.n., 2005.**

32. Piera, Miguel Angel, et al. ***Como mejorar la logística de su empresa mediante la simulación .*** **España : s.n., 2005.**

33. Cuba, Ministerio de Relaciones Exteriores de la República de. ***La informatización en Cuba.*** **2009.**

34. Perez de Alejo, Victoria, Goza León, HE and Pérez Ones, Osney. ***Modelación, Simulación y Análisis de Sisitemas Termoenergéticos de Fabricas de Azúcar Crudo y Refino.*** **CUJAE. Ciudad de la Habana : s.n., 2007.**

35. Villanueva Castillo, José. ***La simulación de procesos, clave en la toma de decisiones .*** **2008.**

36. Taylor, Thomas H. ***Técnicas de simulación en computadora.*** **Limusa : s.n., 1975.**

37. Hernández, Sampieri, Fernández Collado, Roberto and Batista Lucio, Carlos. ***Metodología de la Investigación.*** **México : s.n., 1998.**

38. Farrel, Charles. ***Simulation Made Easy.*** **New York : s.n., 1995.**

39. Team, CAPE-OPEN Project. ***Conceptual Design Document 2.*** **1997.**

40. Domínguez Valente, Carlos and López Octavio, Marco Antonio. ***Simulación Digital.*** **México : s.n., 2008.**

Anexos

Anexo 1 Descripciones de Casos de Uso del sistema

Tabla 10 CUS Obtener Propiedades de Corriente

Caso de Uso:	Obtener propiedades de la corriente	
Actores:	Editor Gráfico	
Resumen:	El Caso de Uso se inicia cuando el Editor Gráfico invoca la función obtener propiedades de las corrientes. El subsistema ejecuta la función invocada y termina el CU.	
Precondiciones:		
Referencias	RF.2, CU incluido: Manipular Unidad de medida.	
Prioridad	Crítico	
Flujo Normal de Eventos		
Acción del Actor		Respuesta del Subsistema
1. El Editor Gráfico invoca la función obtener propiedades de corriente e introduce los datos necesarios.		2. El subsistema comprueba los datos. 3. El subsistema devuelve las propiedades de la corriente.
Flujos Alternos		
Línea 2: El subsistema lanza un mensaje de error porque los datos no son válidos.		
Poscondiciones		

Tabla 11 CUS Identificar Corriente

Caso de Uso:	Identificar corriente.
Actores:	Editor Gráfico
Resumen:	Esta funcionalidad permite identificar las corrientes del proceso.
Precondiciones:	
Referencias	RF6
Prioridad	Crítico

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Subsistema
1. El Editor Gráfico invoca la función identificar corriente e introduce los datos necesarios.	2. El subsistema valida los datos. 3. El subsistema actualiza los datos (nombre e identificador).
Flujos Alternos	
Línea 2: El subsistema lanza un mensaje de error porque los datos no son válidos.	
Poscondiciones	

Tabla 12 CUS Introducir Operación Unitaria

Caso de Uso:	Introducir Operación Unitaria
Actores:	Editor Gráfico
Resumen:	El Caso de Uso se inicia cuando el Editor Gráfico invoca la función introducir corriente de entrada. El subsistema ejecuta y termina el CU.
Precondiciones:	
Referencias	RF.8, RF 8.1, CU incluidos: Evaluar Propiedades
Prioridad	Critico
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Subsistema
4. El subsistema Editor Gráfico invoca la función introducir operación unitaria e introduce los datos necesarios.	5. El subsistema valida los datos. 6. El subsistema crea la nueva operación unitaria con dichos valores.
Flujos Alternos	
Línea 2: El subsistema lanza un mensaje de error porque los datos no son válidos.	
Poscondiciones	

Tabla 13 CUS Evaluar Propiedades

Caso de Uso:	Evaluar propiedades
Actores:	Editor Gráfico
Resumen:	El Caso de Uso se inicia cuando el Editor Gráfico ha modificado las propiedades de una corriente. El subsistema ejecuta y termina el CU.
Precondiciones:	

Referencias	RF.5, RF5.1, RF 5.2
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Subsistema
1. El Caso de Uso se inicia cuando el Editor Gráfico ha modificado las propiedades de una corriente.	2. El subsistema obtiene la presión, temperatura y composición de la corriente que ha sido modificada. 3. El subsistema calcula las propiedades de los componentes asociados a dicha corriente. 4. El subsistema calcula las propiedades dependientes de la corriente modificada. 5. El subsistema modifica las propiedades de la corriente sustituyéndolas por las calculadas.
Flujos Alternos	
Poscondiciones	Quedan actualizadas las propiedades de la corriente modificada.

Tabla 14 CUS Manipular Unidades de Medidas

Caso de Uso:	Manipular Unidades de Medida
Actores:	Editor Gráfico
Resumen:	El Caso de Uso se inicia cuando el Editor Gráfico invoca la función manipular unidades de medida. El subsistema ejecuta y termina el CU.
Precondiciones:	
Referencias	R.F12
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Subsistema
1. El Editor Gráfico invoca la función manipular unidad de medida de una propiedad de un componente e introduce los datos necesarios.	2. El subsistema valida los datos. 3. El subsistema realiza las transformaciones correspondientes en las unidades de medida. 4. El subsistema modifica los valores y las

	unidades de medida.
Flujos Alternos	
Línea 2: El subsistema lanza un mensaje de error porque los datos no están correctos.	
Poscondiciones	

Tabla 15 CUS Introducir Propiedades de Corriente

Caso de Uso:	Introducir propiedades de corriente
Actores:	Editor Gráfico
Resumen:	El Caso de Uso se inicia cuando el Editor Gráfico invoca la función introducir propiedades de operación unitaria. El subsistema ejecuta y termina el CU.
Precondiciones:	
Referencias	R.F11 CU incluido: Manipular Unidad de medida, Evaluar Propiedades
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Subsistema
1. El Editor Gráfico invoca la función introducir propiedades de operación unitaria e introduce los datos necesarios.	2. El subsistema valida los datos. 3. El subsistema modifica las propiedades de la corriente (presión, temperatura o composición). 4. Ejecuta CU Evaluar propiedad.
Flujos Alternos	
Línea 2: El subsistema lanza un mensaje de error porque los datos no están correctos.	
Poscondiciones	

Tabla 16 CUS Salvar Resultado

Caso de Uso:	Salvar Resultado
Actores:	Editor Gráfico
Resumen:	El Caso de Uso se inicia cuando el Editor Gráfico invoca la función salvar resultado. El subsistema ejecuta y termina el CU.
Precondiciones:	
Referencias	R.F 13
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Subsistema

1. El Editor Gráfico invoca la función salvar resultado.	2. El subsistema salva la información utilizando ficheros.
Flujos Alternos	
Poscondiciones	

Tabla 17 CUS Determinar Orden de Cálculo

Caso de Uso:	Determinar Orden de Cálculo
Actores:	Editor Gráfico
Resumen:	El Caso de Uso se inicia cuando el Editor Gráfico invoca la función determinar orden de cálculo. El subsistema ejecuta y termina el CU.
Precondiciones:	
Referencias	R.F10
Prioridad	Crítico
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Subsistema
1. El Editor Gráfico invoca la función determinar orden de cálculo e introduce los datos necesarios.	2. El subsistema valida los datos. 3. El subsistema determina el orden en que deben ser ejecutadas las operaciones unitarias en el proceso.
Flujos Alternos	
Línea 2: El subsistema lanza un mensaje de error porque los datos no están correctos.	
Poscondiciones	

Anexo 2 Diagramas de Secuencia

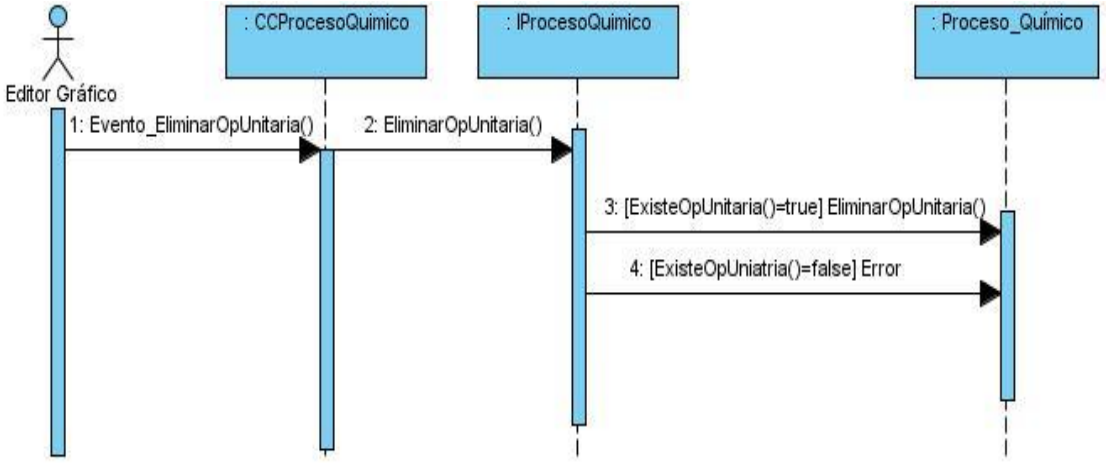

Ilustración 25 DS Gestionar Proceso (Eliminar Operación Unitaria)

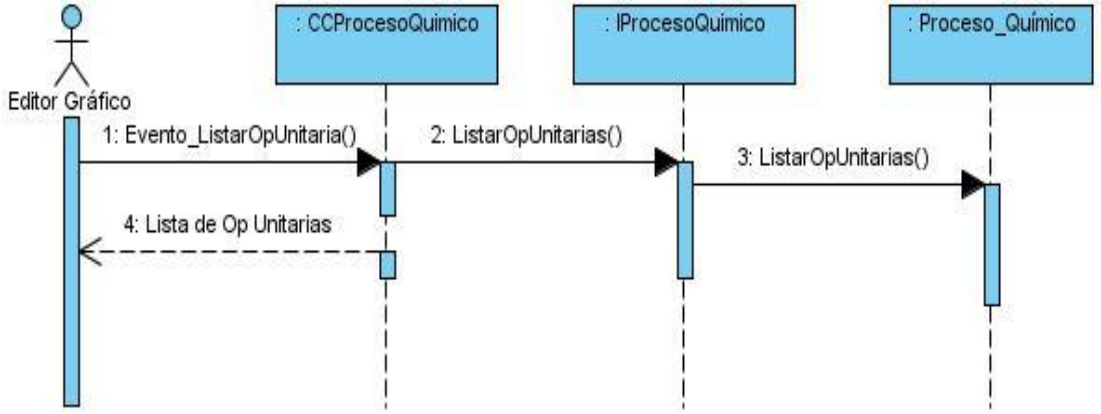

Ilustración 26 DS Gestionar Proceso (Listar Operaciones Unitarias)

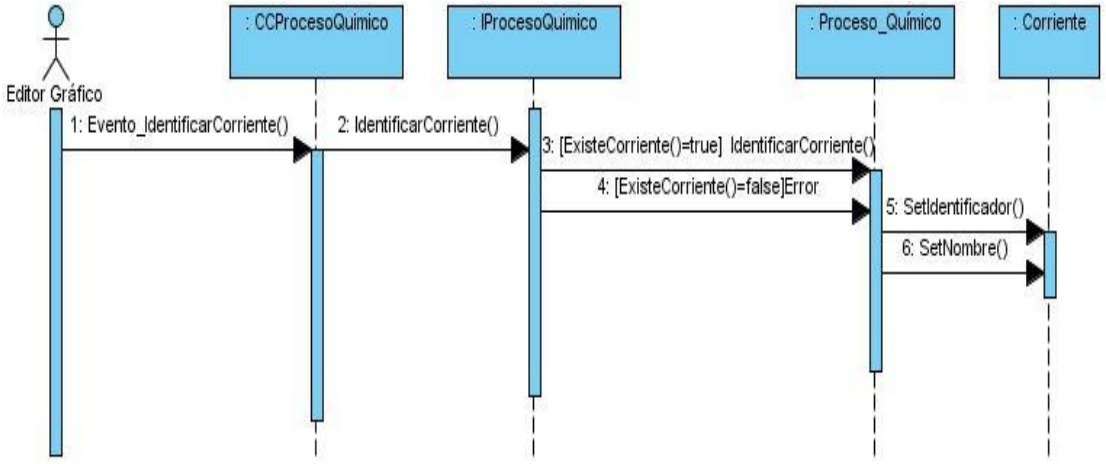

Ilustración 27 DS Identificar Corriente

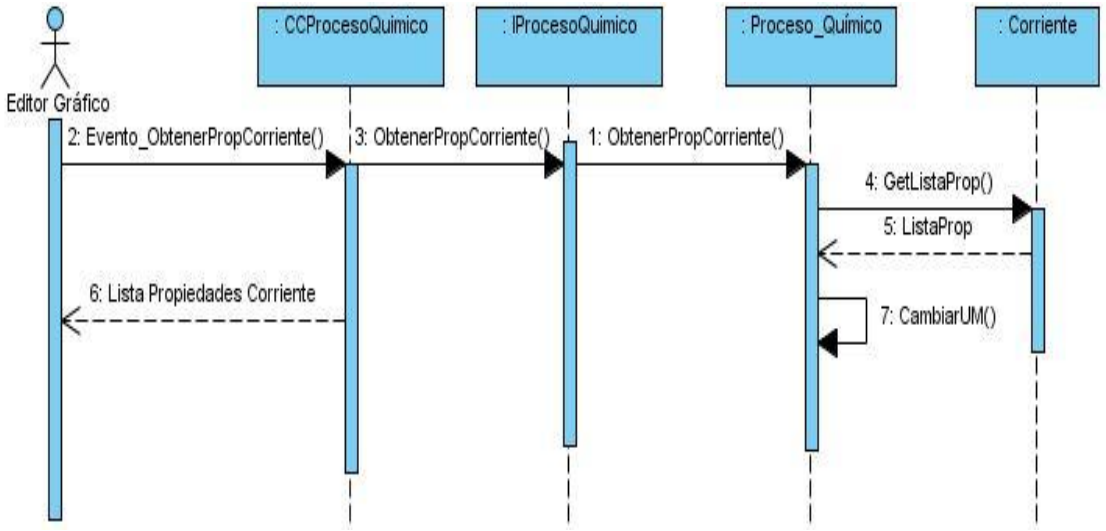

Ilustración 28 DS Obtener Propiedades de Corriente

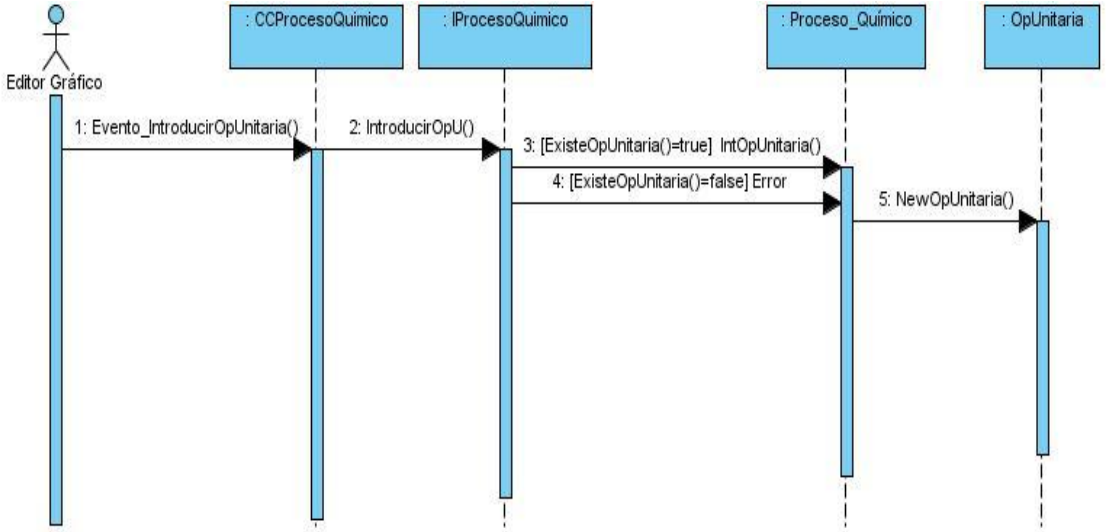

Ilustración 29 DS Introducir Operación Unitaria

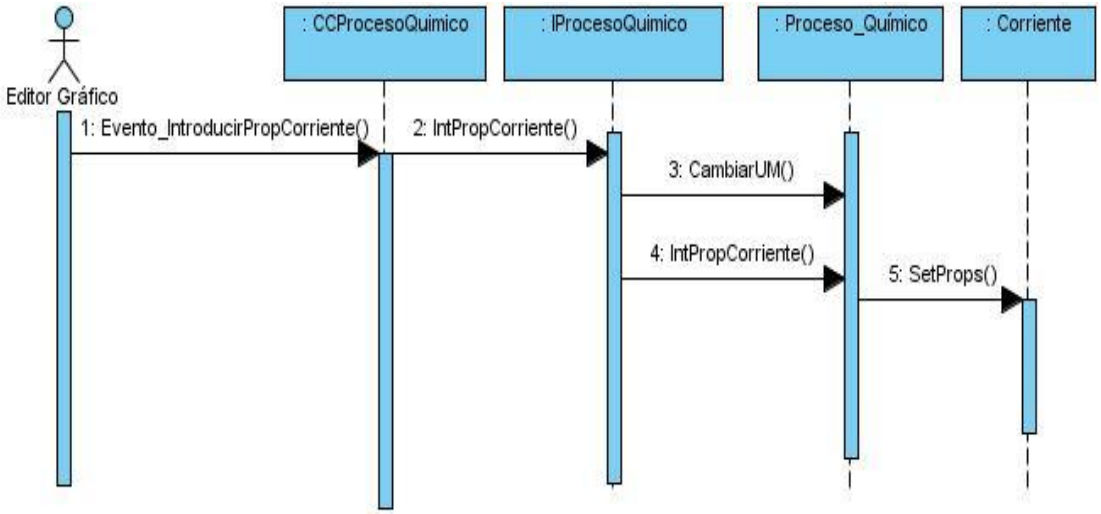

Ilustración 30 DS Introducir Propiedades de Corriente

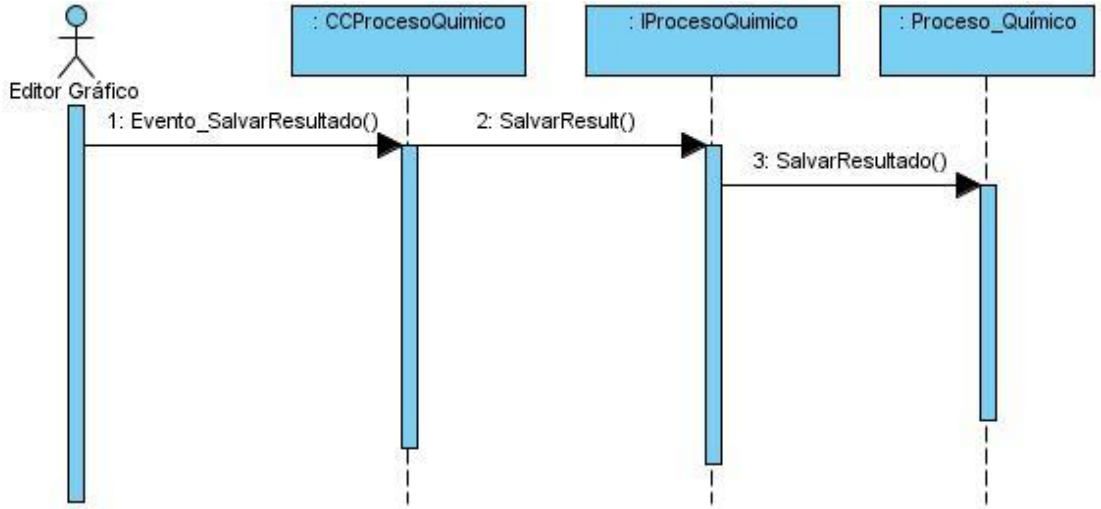

Ilustración 31 DS Salvar Resultado

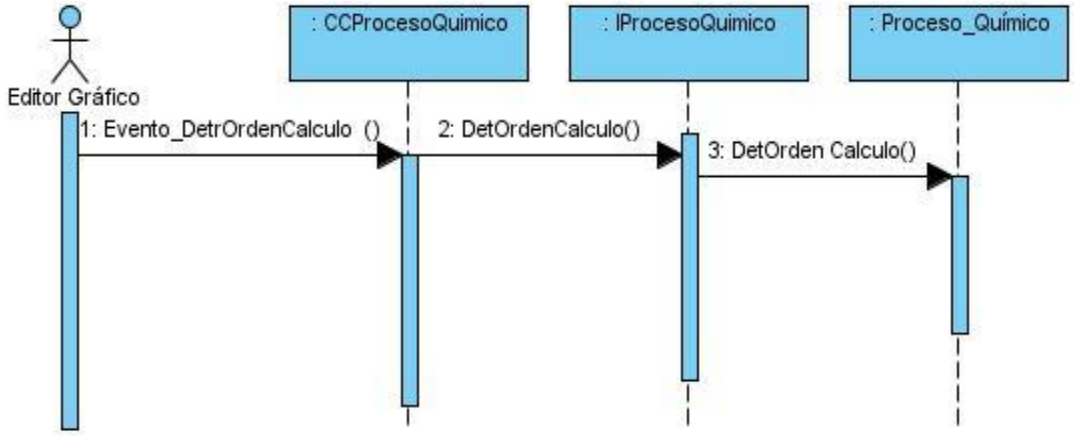

Ilustración 32 DS Determinar Orden de Cálculo

Glosario de Términos

Simulación: Es una técnica matemática que imita el funcionamiento de un proceso del mundo real en un tiempo determinado, con el propósito de explicar, entender o mejorar el proceso.

Proceso Químico: Es el conjunto de operaciones químicas que recorre un producto desde la adquisición de la materia prima hasta lograr el producto final.

Simulación de Procesos: Desarrollar modelos matemáticos que se concentren en los aspectos principales del proceso y que calculados repetida y creativamente permitan identificar las principales tendencias del comportamiento del proceso.

Operación unitaria: Se refiere a una abstracción de los equipos reales de la industria, cada operación unitaria en el sistema tiene correspondencia con un equipo de la vida real, a excepción de los módulos lógicos que fueron incluidos para simular otros aspectos de la industria.

Puertos: Se refiere a los puntos que permiten la conexión entre operaciones unitarias, los puertos pueden ser de entrada o salida.

Propiedades físico químicas: Conjunto de propiedades ya sean físicas o químicas que representan las características de un componente químico determinado.

Componentes Químico: Sustancia química involucrada en el proceso.

Corriente: Se refiere a los flujos de distintas sustancias que existen en la industria, por ejemplo, en el caso de la industria azucarera existen corrientes de jugo, meladura, vapor, agua, etc.

DFI: Diagrama de Flujo de Información, se refiere a los diagramas que dibujan los ingenieros químicos u otros especialistas que modelan la estructura de la fábrica que se desea simular.

Biblioteca de Clases: Una biblioteca es un conjunto de subprogramas utilizados para desarrollar software. Las bibliotecas contienen código y datos, que proporcionan servicios a programas independientes, es decir, pasan a formar parte de éstos. En el lenguaje de programación Java, las bibliotecas están típicamente empaquetadas en ficheros JAR y pueden contener varias clases diferentes de objetos. El tipo más importante de objeto contenido en un fichero JAR es una clase Java.