Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas .

Título: Componente para la configuración y manejo de recetas del SCADA GALBA

Autor: Yosvany Leyva Pizarrosa

Tutores: Ing. Oscar Calderín Pérez

Co-tutores: Ing. Elizabeth Díaz Martínez

Ing. David Gómez García

La Habana, junio de 2015 .

"Año 57 de la Revolución"

Declaración de Autoría

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Yosvany Leyva Pizarrosa

_____

Firma del Autor

Ing. Oscar Calderín Pérez

_____

Firma del Tutor

Ing. David Gómez García

_____

Firma del Co-tutor

Ing. Elizabeth Díaz Martínez

_____

Firma del Co-tutor

Datos de contacto

Datos del Tutor

Ing. Oscar Calderín Pérez
Institución: Universidad de las Ciencias Informáticas (**UCI**).
Título: Ingeniero en Ciencias Informáticas
e-mail: ocalderin@uci.cu

Graduado en la UCI en el año 2012, especialista B con 3 años de experiencia en el desarrollo de software.

Datos de los Co-tutores

Ing. Elizabeth Díaz Martínez
Institución: Universidad de las Ciencias Informáticas (**UCI**).
Título: Ingeniero en Ciencias Informáticas
e-mail: edmartinez@uci.cu

Graduado en la UCI en el año 2014, actualmente se encuentra en adiestramiento

Ing. David Gómez García
Institución: Universidad de las Ciencias Informáticas (**UCI**).
Título: Ingeniero en Ciencias Informáticas
e-mail: dggarcia@uci.cu

Graduado en la UCI en el año 2014, actualmente se encuentra en adiestramiento

Agradecimientos

Agradezco a todos los que de una forma u otra contribuyeron al logro de este gran sueño, convertirme en ingeniero. A mis familiares, en especial a mi madre, mi hermano, mi padre y mi abuelo. A mi tías, especialmente a Vilma, mi madre durante estos 5 cursos por todo su amor, paciencia y dedicación.

A mi novia por todo su amor, apoyo y confianza, a mis compañeros de apto, especial al piquete a Yugui, Mota, Nelseker, Susano, el Niche, el White, Pedro. A mis compañeros de grupo, especial a mi amiga Marelis que desde una rivalidad muy sana en cuanto a quien salía mejor, me ayudo a dar mis mejores esfuerzos. A mis compañeros del Movimiento de programación competitiva, a Tommy, al Dovi, mis compañeros de equipo.

A mis tutores Oscar, Arian, mis cotutores David y Elizabeth por todas las horas de sueño empleadas en ayudarme. A mis amigas las analistas del laboratorio 22, las Yadiras y Yanet.

Dedicatoria

A mis padres, mi familia, mis amigos y a todos los que de una forma u otra me brindaron ese apoyo tan necesario para lograr este gran sueño y poderme convertir en lo que soy hoy, este trabajo también es de ustedes.

A mi madre, hermano, a mis abuelos y en especial a mi abuelo Luis que no se cansó de repetirme que en la vida todo merece empeño y sacrificios, todos mis logros como estudiante se los debo a ellos.

A mis amistades por todos momentos compartidos.

Resumen

Un sistema de Supervisión, Control y Adquisición de Datos (**SCADA**) se encarga de monitorear los eventos que tienen lugar en estaciones remotas. La Universidad de las Ciencias Informáticas (**UCI**) de Cuba, específicamente el Centro de Informática Industrial (***CEDIN***) perteneciente a la facultad 5 es protagonista del desarrollo de un sistema de este tipo conocido como SCADA Guardián del Alba (**GALBA**). Debido a la gran complejidad de estos sistemas, su desarrollo fue dividido en módulos. Entre los distintos módulos que lo componen se encuentra el de Aplicaciones, el cual se encarga de agregarle al sistema distintas aplicaciones que le permitan un mayor control de los procesos industriales.

En la actualidad el SCADA GALBA carece de mecanismos que le permitan emplear estrategias y soluciones automatizadas para dar respuesta a tareas (**procesos por lotes**) que cumplen con un número específico de condiciones y especificaciones dadas. Por tal motivo con el presente Trabajo de Diploma se pretende dotar al GALBA, específicamente al Planificador de tareas del módulo Aplicaciones de una herramienta para la configuración y gestión de recetas, las cuales van a permitir en lo adelante al SCADA GALBA dar cumplimiento a las tareas antes referidas.

Palabras claves: automatización de procesos**,** GALBA, proceso por lotes, recetas, SCADA.

Índice

Índice de Figuras

Índice de Tablas

Introducción

El hombre desde sus propios inicios ha transitado por varias etapas de gran influencia para lograr el desarrollo científico-técnico y social alcanzado por la sociedad actual, marcando pautas en grandes ramas de las ciencias como las matemáticas, la física, la salud y la astronomía por solo citar algunas. Otro campo muy explorado por el hombre lo es la automatización de las procesos industriales asociados a su propia existencia, uno de los avances más significativos lo constituye el surgimiento de los Sistemas **SCADA** (en inglés *Supervisory Control And Data Acquisition*) como herramienta para la supervisión, el control y la adquisición de los datos resultantes de dichos procesos. En la actualidad los altos índices de consumos alcanzando por la humanidad han llevado a un aumento sustancial de la producción y por ende a la búsqueda de nuevas alternativas para hacer frente a dicho aumento, la implantación de los sistemas ***batch*** o procesos por lotes se ha destacado como una de las propuesta ante esta situación. Se conoce como sistema por lotes (en inglés *batch processing*), o modo batch, a la ejecución de un programa sin el control o supervisión directa del usuario (que se denomina proceso interactivo).[1] Este tipo de programas se caracterizan porque su ejecución no precisa ningún tipo de interacción con el usuario.

Las industrias de producción que emplean estrategias y soluciones automatizadas han demostrado ser el ejemplo a seguir para la elaboración de una serie de productos que cumplen con un número específico de condiciones y especificaciones dadas. Una rama significativa de estas empresas lo constituyen las dedicadas a la industria petrolífera. Un aspecto en común entre estas empresas es la producción por lotes o recetas (*recipes*) (diversidad de mezclas) resultantes de los procesos de extracción, transporte y almacenamiento; además de la diversidad de productos resultantes del proceso de refinado del hidrocarburo. Gracias al concepto de receta[1] es posible para los sistemas SCADA almacenar y recuperar paquetes de datos que permiten configurar un sistema de forma automática. De esta manera, el procedimiento de cambiar la configuración de trabajo de toda una planta de proceso quedará reducido al simple hecho de pulsar un botón después de confirmar unos datos de acceso (usuario, contraseña y número o nombre de receta). El sistema SCADA se encargará de enviar los datos a los correspondientes

1 Se tratan de archivos que guardan los datos de configuración de los diferentes elementos del sistema (velocidad de proceso, presiones, temperaturas, niveles de alarma, cantidades de piezas, entre otros.).

controladores, quedando la planta lista para las nuevas condiciones de trabajo reduciendo al máximo los errores que puedan ingresarse al sistema; producto del factor humano. El empleo de recetas aplicado al campo de la seguridad, puede automatizar aún más el proceso, de manera que, ante una situación imprevista, se ejecute una determinada receta que coloque a los diferentes elementos en una posición no comprometida, evitando así posibles daños al proceso o a sus componentes.[2]

En la actualidad existen diferentes SCADA con grandes potencialidades; en su gran mayoría privativos, compañías como *Rockwell Automation*, *Wonderware, COPA-DATA* y *Siemens* lideran a nivel mundial el mercado en lo referente a estos sistemas, es por ello que a raíz del paro petrolero sufrido por Venezuela en el año 2006 y buscando alcanzar la soberanía tecnológica en el campo de la automatización de la industria del petróleo de Venezuela, surge el SCADA Guardián del Alba (***GALBA***) en agosto de este propio año como proyecto de colaboración entre las hermanas repúblicas de Cuba y Venezuela. La Universidad de las Ciencias Informáticas (**UCI**) de Cuba, es protagonista del desarrollo de este sistema, específicamente el Centro de Informática Industrial (***CEDIN***) perteneciente a la facultad 5. El SCADA Guardián del Alba ha demostrado ser una buena opción a la supervisión, el control y la adquisición de los datos resultantes de la industria del petróleo en Venezuela, pero el mismo en estos momentos carece de mecanismos que le permitan emplear estrategias y soluciones automatizadas para dar respuesta a tareas que cumplen con un número específico de condiciones y especificaciones dadas (**los procesos por lotes**) los cual son de una importancia relevante en aras de hacer del GALBA un SCADA competitivo dentro del mercado de estos sistemas, así como alcanzar los índices de producción demandados por la sociedad actual. El empleo de estos procesos es una práctica muy común en las industrias para las cuales la dosificación, mezcla y transporte de material constituyen pasos de primer orden en su proceso productivo. Es por ello que la demanda de estos procesos en las empresas dedicas a la extracción y refinamiento de petróleo es de uso frecuente por la diversidad de grado o gravedad API[2] presente en los petróleos crudos, lo que obliga a que las condiciones de trabajo de una planta sean cambiantes constantemente en dependencia del tipo de material con el que trabaja. Por un oleoducto se envían petróleos y derivados de diferentes categorías por lo que se hace necesario para su tratamiento adaptar las condiciones del mismo en dependencia del tipo de material que se transporta es por ello que, los

2 Siglas en inglés de American Petroleum Institute.

mismo son tratados mediante lotes de producción. El GALBA dentro sus funciones incluye el control de los procesos transporte y almacenamiento del petróleo extraído en Venezuela, por lo que se hace necesario en el mismo la inclusión de mecanismos para la gestión de los procesos por lotes. Durante la presentación y el despliegue del SCADA GALBA es muy señalada la imposibilidad del sistema para gestionar los procesos por lotes o *batch*, situación que provoca que que el GALBA no pueda ser instalado donde se gestionan varios procesos en el mismo nivel, motivo que impulsa al desarrollo de la siguiente investigación.

Como resultado de lo analizado anteriormente surge el planteamiento del siguiente problema científico: ¿Cómo gestionar los procesos por lotes en el SCADA GALBA? a partir del problema científico antes planteado se puede definir como objeto de estudio: la gestión de los procesos por lotes en los Sistemas SCADA. Tomando como punto de referencia el objeto de estudio definido anteriormente se propone como campo de acción: la gestión de procesos por lotes en los sistemas SCADA mediante recetas básicas.

El objetivo general de la presente investigación consiste en desarrollar un componente que permita gestionar los procesos por lotes mediante recetas.

Para desarrollar satisfactoriamente la investigación se han trazado las siguientes **tareas investigativas:**

- Elaboración el marco teórico de la investigación científica a partir del estado del arte actual del tema a investigar.
- Caracterización de la metodología, herramientas y tecnologías a utilizar en el desarrollo del componente.
- Identificación de los requisitos funcionales y no funcionales para la creación del componente.
- Análisis y diseño del sistema.
- Implementación de la solución propuesta**.**
- Validación de la solución propuesta.

Durante el proceso de investigación se tendrán en cuenta los siguientes métodos científicos los cuales resultan muy necesarios para la elaboración del presente trabajo.

Métodos Empíricos:

- **Consulta de fuentes de Información:** Este método se empleó en la consulta de las fuentes de información necesarias para el desarrollo de la investigación.

- **Observación:** Este método se utilizó para apreciar el comportamiento del componente desarrollado de acuerdo a los algoritmos implementados en el mismo.

- **Pruebas:** Este método se empleó para la validación de los resultados obtenidos con la solución propuesta a partir de la investigación realizada.

Métodos Teóricos:

- **Análisis histórico-lógico:** Utilizado para conocer, con mayor profundidad los antecedentes y las tendencias actuales de las herramientas y tecnologías.

- **Modelación**: Este método se empleó para definir y representar gráficamente las funcionalidades del sistema usando el Lenguaje Unificado de Modelado(**UML**).

Para una mayor organización el presente trabajo se ha estructurado en **3** capítulos de los cuales se muestra una pequeña síntesis a continuación:

Capítulo 1: Fundamentación Teórica.

En este capítulo se realiza un pequeño bosquejo de la propuesta y un estudio del estado del arte sobre los sistemas SCADA, sus principales características y funcionalidades, sus estrategia y mecanismos para responder a los procesos por lotes. Seguidamente se realiza la selección de las tecnologías y metodologías de desarrollo a utilizar en la implementación y desarrollo de la solución. Por último se reflejan mediante las conclusiones la posición del autor acerca de los temas analizados.

Capítulo 2: Análisis y diseño de la Herramienta

En este capítulo se presenta la solución propuesta mediante una descripción de las funcionalidades

seleccionadas para el desarrollo y de las características que se conciben para el sistema. En dicho capítulo se elaboran las historias de usuarios correspondientes, las cuales guiarán el desarrollo de forma organizada, se modela la solución propuesta usando la metodología de desarrollo seleccionada, además de explicar los patrones y arquitectura seleccionados para la implementación de la solución.

Capítulo 3 Implementación y Prueba de la Herramienta

En este capítulo se detallan las tres iteraciones realizadas durante la etapa de implementación, definiendo las funcionalidades como tareas de ingeniería de acuerdo a lo planteado por la metodología de desarrollo escogida. Se definen las pruebas a realizar al componente, especificando los resultados obtenidos mediante las pruebas de aceptación realizadas para obtener la valoración de dichas pruebas de manera cualitativa.

Capitulo 1. Fundamentación Teórica

En el presente capítulo se introducen las principales características y conceptos asociados a los sistemas SCADA, las normas y estándares que rigen la automatización de procesos, así como la forma de producción por lotes empleadas por el hombre en la automatización de procesos. Para ello se abordan definiciones, importancia, ventajas y desventajas de esta forma de producción. Finalmente, con le fin de conformar la solución técnica del sistema se seleccionaron las herramientas y tecnologías para el desarrollo de la solución.

1.1 Sistemas SCADA

En los últimos años el crecimiento de las industrias ha traído consigo un gran desarrollo y extensión de las fábricas y maquinarias que la componen, lo que ha conllevado a un aumento en la complejidad de los procesos industriales. Para lograr un control eficiente de estos procesos surgieron los sistemas de supervisión, control y adquisición de datos, conocidos como SCADA. Se trata de una aplicación de software especialmente diseñada para funcionar sobre computadoras en el control de producción, proporcionando comunicación con los dispositivos de campo (controladores autónomos, autómatas, programables, entre otros) y controlando el proceso de forma automática desde la pantalla de la computadora. Además dicho sistema provee de toda la información que se genera en el proceso productivo a diversos usuarios, tanto del mismo nivel como de otros supervisores dentro de la empresa: control de calidad, supervisión, mantenimiento, entre otros.

1.1.1 Principales funcionalidades de un SCADA

Un sistema SCADA como parte integral de la estructura de las industrias y un recurso importante de información debe cumplir con tres funciones principales: [3]

- **Adquisición de datos** para recoger, procesar y almacenar la información recibida.
- **Supervisión** para observar desde el monitor la evolución de las variables del proceso.
- **Control** para modificar la evolución del proceso, actuando sobre los reguladores autónomos básicos (consignas, alarmas, menús, entre otros.), o directamente sobre el proceso mediante las

salidas conectadas.

1.1.2 SCADA GALBA

La UCI en conjunto con la Gerencia Automatización Informática y Telecomunicaciones (**AIT**) de la empresa Petróleos de Venezuela S.A., desarrollan desde el año 2006 un proyecto de software con el objetivo de crear un producto SCADA, conocido como SCADA Nacional o SCADA PDVSA en sus inicios, entre 2006 y 2007, y SCADA GALBA luego de ser presentado en la "Cumbre del ALBA" en enero de 2008, proyectando una futura instalación del sistema en los países integrantes de esta organización.

El GALBA es un sistema distribuido en módulos que trabajan de manera conjunta posibilitando el funcionamiento del sistema como un todo. Estos módulos se encuentran interconectados a través de un software para la distribución de los servicios en la red (***Middleware***), conocido como software de comunicación entre aplicaciones. La distribución de los módulos existentes en el SCADA permite obtener configuraciones escalables en dependencia de los requisitos que presente cada módulo. Es un sistema en tiempo real y presenta una arquitectura distribuida. Está dividido en varios subsistemas entre los que se encuentran:[4]

- Comunicación.
- Adquisición.
- Configuración.
- Almacenamiento de datos históricos.
- Seguridad.
- Visualización o HMI.
- Aplicaciones.

1.2 Automatización Industrial

La automatización industrial es la aplicación de diferentes tecnologías para controlar y monitorear un proceso, máquina, aparato o dispositivo que por lo regular cumple funciones o tareas repetitivas,

haciendo que opere automáticamente, reduciendo al mínimo la intervención humana.[5] Su principal objetivo es generar la mayor cantidad de productos en el menor tiempo posible, con el fin de reducir los costos y garantizar una uniformidad en la calidad. En el enfoque de la industria, la automatización es el paso más allá de la mecanización en donde los procesos industriales son asistidos por máquinas o sistemas mecánicos que reemplazan las funciones que antes eran realizadas por animales. Mientras en la mecanización los operadores son asistidos con maquinaria a través de su propia fuerza y de su intervención directa, en la automatización se reduce de gran manera la necesidad mental y sensorial del operador.

1.2.1 Producción por Lotes

Es el sistema de producción que emplean las empresas que producen una cantidad limitada de un producto cada vez, al aumentar las cantidades más allá de las pocas que se fabrican al iniciar la compañía, el trabajo puede ser realizado de esta manera. Esa cantidad limitada se denomina lote de producción. Estos métodos requieren que el trabajo relacionado con cualquier producto se divida en partes u operaciones, y que cada operación quede terminada para el lote completo antes de emprender la siguiente operación. Esta técnica es tal vez el tipo de producción más común. Su aplicación permite cierto grado de especialización de la mano de obra, y la inversión de capital se mantiene baja, aunque es considerable la organización y la planeación que se requieren para librarse del tiempo de inactividad o pérdida de tiempo.[6] Entre las ventajas más significativas de este modelo de producción están:

- Pueden reducir los costos iniciales de establecimiento porque una sola cadena de producción se puede utilizar para fabricar diferentes productos.
- Resulta una alternativa a empresas que no pueden llevar a cabo producción continua, permite a un producto cesar su producción sin asumir grandes pérdidas.

1.3 Norma ISA 88

La integración de los procesos resultantes de las industrias que emplean soluciones automatizadas en sus procesos de producción han constituido un problema al que numerosas compañías y empresas se han tenido que enfrentar durante años, es por ello que desde inicios de la década de los 90 una

importante institución en la rama de la automatización, como lo es la Sociedad Internacional de Automatización (**ISA**) por sus siglas en inglés, observando esta dificultad, sus especialistas aunaron esfuerzos en la búsqueda de una solución. Es así como surge el estándar ANSI/ISA 88 como protocolo de comunicación para permitir la unión de los sistemas de gestión con el nivel de planta.[7] En términos sencillos la norma ISA 88 desarrolló una metodología *batch* que permite a las industrias crear formas estándares para automatizar la producción *batch* y al mismo tiempo, reducir tanto la complejidad como los costos asociados a los sistemas propietarios.

1.3.1 Modelos de la ISA 88

Para un mejor entendimiento de lo que la norma ISA 88 define en términos de receta, este documento se referirá brevemente a los modelos que se interrelacionan para lograr la gestión de procesos batch con el uso de recetas. Dentro de estos modelos se encuentran:

Modelo Físico: Propone una organización de los componentes que forman una empresa y todo su equipamiento tecnológico. Como resumen un sitio está formado por áreas las cuales contienen unidades de control, equipamiento de control y finalmente módulos de control. [8]

Modelo de Proceso: Propone organizar un proceso en varias etapas de proceso, cada etapa se compone de varias operaciones de proceso y a su vez cada operación está compuesta por acciones específicas. [8]

Modelo de Procedimientos o Procedural: Describe las relaciones entre los procedimientos, las tareas que lo componen para cada equipamiento.[8]

1.3.2 Relación entre modelos

La figura 1 muestra la relación entre los diferentes modelos propuestos por ISA 88

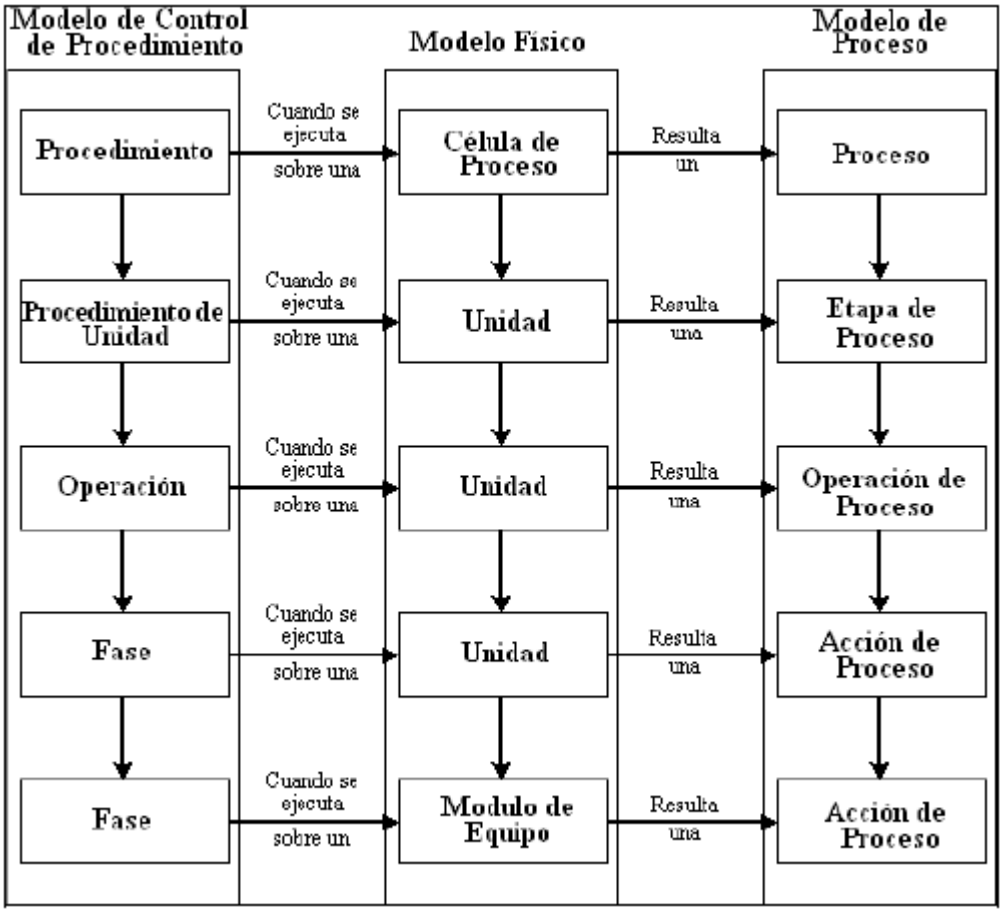

Figura 1: Relación entre modelos de la Norma ISA 88.

Como resumen se puede plantear que en una aplicación las entidades equipos pueden realizar funciones definidas en los procedimientos. Mientras los elementos de procedimientos son definidos de manera que se correspondan con elementos del modelo de proceso. Es importante resaltar que el modelo de procedimiento puede estar implícito como parte del equipamiento o definido como parte de una receta. Los equipos pueden contener elementos que implementen procedimientos de control. La combinación del equipo de control y los elementos procedurales definidos para ese equipo se refieren a la capacidad de ese equipo y puede también estar definidas en la recetas.

1.4 Recetas

La necesidad de obtener productos de calidad en ambientes flexibles en cuanto a mantener múltiples condiciones de proceso u obtener múltiples productos con el mismo equipamiento tecnológico dio paso, en el control de procesos actual al concepto de receta. Una receta en el control de procesos, tiene los mismos conceptos funcionales de una receta de cocina como concepto general se puede resumir como un conjunto de información que define las cantidades de material, condiciones y procedimientos necesarios para obtener un producto con un equipamiento determinado.[8] Una de las normas más utilizadas en el empleo de recetas en el control de procesos lo es la ISA 88, especializada en la gestión de control de procesos con enfoques *batch.*

1.4.1 Recetas definidas por ISA 88

Según ISA 88 las recetas son una colección de información que define los requerimientos para la manufactura de un producto específico, estas proporcionan una manera de describir los productos y cómo esos productos pueden ser producidos.[9] Dependiendo de los requisitos específicos de una empresa pueden existir otros tipos de recetas. Sin embargo esta norma solo aborda 4 tipos de recetas: **Receta General**, **Receta de Sitio**, **Receta Maestra y Receta de Control**.

La figura 2 muestra las relaciones entre los 4 tipos de recetas propuestas por ISA 88. La **Receta General** es aplicable en el ámbito de la empresa y sirve como base para el nivel más bajo de recetas. Esta incluye solo información específica del producto y su procesamiento sin incluir ninguna información sobre el equipamiento ni la fábrica en la cual se va a ejecutar. Las cantidades que se obtienen en laboratorios se expresan en términos normalizados y se agregan las restricciones y capacidades que debe tener el equipamiento en el cual se vaya a fabricar. Además no contiene ninguna información referida a la producción, como planificación ni clientes finales.

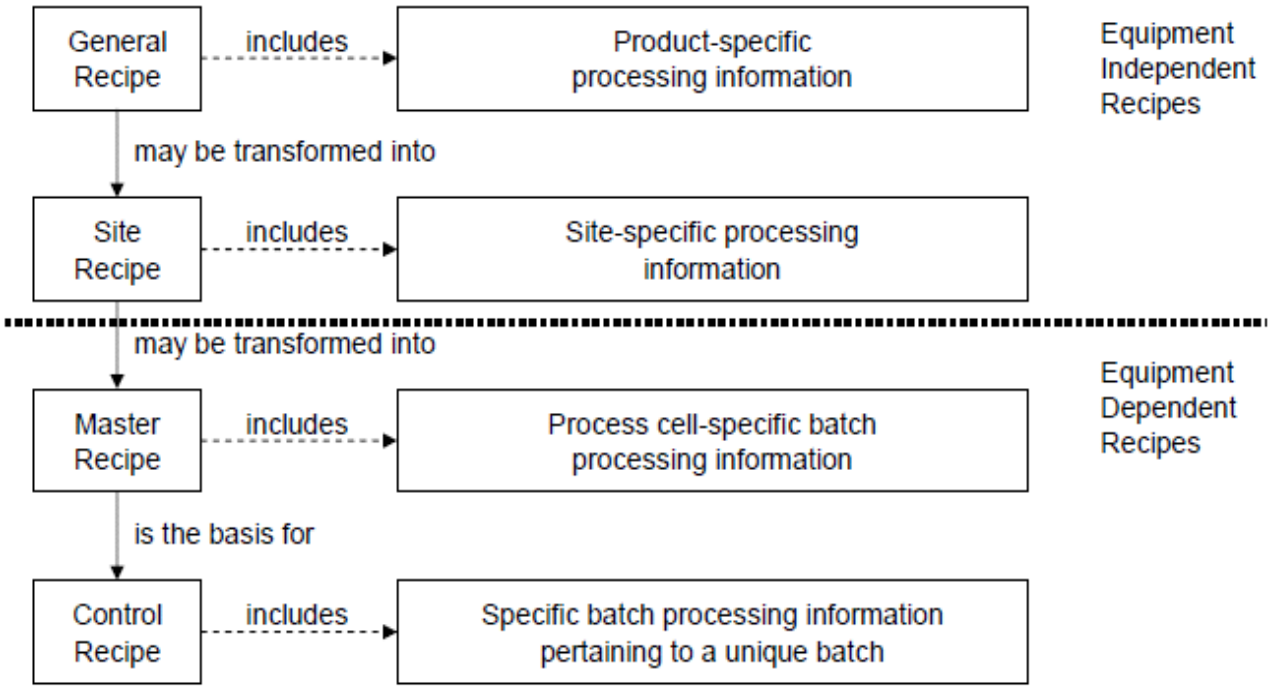

Figura 2: Relación entre tipos de recetas según ISA 88. Tomada de la especificación original

La **Receta de Sitio** se deriva de la **Receta General**, la misma agrega información específica de la fábrica en la que se va a obtener el producto sin especificar equipamiento. En ella se realizan conversiones a la **Receta General** para normalizar los conceptos a los estándares de la fábrica, nomencladores, unidades de ingeniería, se agregan nuevas restricciones basados en el equipamiento existente.[8]

A partir de cualquiera de las recetas de alto nivel (**General** o **de Sitio**) se puede obtener la **Receta Maestra**, aunque este tipo receta puede crearse sin necesidad de las anteriores. Esta receta ya incluye la información de en qué equipamiento va a ejecutarse cada acción definida en los procedimientos, las cantidades son normalizadas o valores calculados expresados como ecuaciones.

Finalmente se encuentra la **Receta de Control** que es la que se deriva a partir de la Maestra, teniendo en cuenta el pedido específico y las cantidades normalizadas de la **Receta Maestra**, su contenido define un solo *batch.* Puede modificarse en cualquier momento y tiene incluido la planificación de la producción.

1.4.2 Recetas Básicas

Las **Recetas Básicas** se derivan de las recetas ISA 88(**General y de Sitio**), concentrando su contenido sólo en la formulación y obviando la parte procedural. Este tipo de receta es muy utilizada en sistemas SCADA que deleguen los procedimientos de la recetas en los dispositivos de control, limitándose a la modificación de las condiciones de proceso.[8] Este concepto se incluye en este documento con la idea de describir una opción del uso de recetas en sistemas no necesariamente tipo *batch* como especifica ISA 88. Las Recetas Básicas se caracteriza fundamentalmente por no contener ningún tipo de Procedimiento de Receta, sino su información fundamental y su fórmula. Las Recetas Básicas limitan la información que contiene la receta a las condiciones de procesos necesarias en un momento determinado.[8] Por tanto una receta básica fundamentalmente está compuesta por:

- Información general de la receta o encabezado simplificado.
- Fórmula simplificada.

1.5 Recetas en los Sistemas SCADA

En el presente epígrafe se analizan algunos sistemas SCADA que en la actualidad hacen uso de las recetas para la automatización de sus procesos por lotes. Se analizarán estos software para estudiar la forma en que hacen uso de las recetas y para exponer las ventajas que ha traído consigo el uso de las mismas y tomar el resultado de este estudio como punto de partida del presente trabajo.

1.5.1 COPA-DATA Zenon

Zenon es un software de automatización industrial **abierto**, orientado a objetos. Son muchas las empresas que utilizan la familia de productos de Zenon en todo el mundo como pantalla de operador (HMI) y de control de procesos (SCADA) para visualizar sus procesos. Su diseño abierto permite conectarlo de forma rápida y eficiente con todos los hardware y software que se desee.[10]

Este en su versión 7 incluyó un módulo para la gestión de procesos por lotes siguiendo las pautas definidas por la norma ISA-88. La flexibilidad de la solución en su conjunto es el resultado de la separación entre el control de equipos y el control de ejecución de fórmulas. Esto significa que es posible introducir cambios en la fórmula de un lote sin variar la automatización de equipos y en general, sin

ninguna otra modificación técnica. El control de lotes en Zenon permite crear y gestionar fórmulas maestras, así como plantillas para las fórmulas que se utilizan durante la producción y que reciben el nombre de fórmulas de control. Los procedimientos, como componente principal de una fórmula, están diseñados gráficamente por las personas responsables del producto. Dependiendo de la complejidad del proceso, es posible elegir entre dos representaciones distintas: matriz o gráfico de función del proceso.[11] La ejecución de una fórmula en el control de lotes de Zenon ofrece un gran número de mecanismos para mantener el proceso bajo control, pero también para adoptar las acciones oportunas en caso de desviación del proceso correcto.

Entre las principales ventajas de la inclusión del módulo para la gestión de procesos por lotes se encuentran:

- Flexibilidad económica para la producción por lotes, basada en el cumplimiento de la norma ISA-88.
- Alto grado de conectividad con equipos de producción nuevos o existentes.
- Facilidad de uso mediante parametrización.
- Control fiable mediante un motor de ejecución basado en un modelo de estados y mecanismos de propagación de estados.

1.5.2 Simatic WinCC SCADA

Simatic WinCC es un sistema de visualización de procesos escalonado y dotado de potentes funcionalidades para la supervisión de procesos automatizados.[12] Siemens añadió la opción WinCC/SES **v7.3** (sistema de ejecución de la secuencia) a su Simatic WinCC SCADA (control de supervisión y adquisición de datos) del sistema para el control secuencial de las operaciones basadas en recetas y en secuencias en plantas de producción.

Esta incorporación es ideal para instalaciones cuya dosificación, mezcla y transporte de material son pasos importantes del proceso, como por ejemplo en la industria alimentaria y de bebidas. Gracias al control secuencial flexible, los operadores de plantas son capaces de definir sus etapas de producción de manera clara y combinarlo libremente en secuencias individuales del proceso. Ambas secuencias de

pasos y los parámetros se pueden adaptar en línea en cualquier momento para satisfacer los requisitos de producción en curso.[13]

Esto significa que las secuencias de producción se pueden volver a ajustar de forma rápida y sencilla en el caso de que la calidad de las materias primas naturales fluctúa o si es necesario otra secuencia de pasos de producción para habilitar el enrutamiento flexible a través del sistema de producción. Además, Simatic WinCC / SES V7.3 ofrece un alto nivel de disponibilidad del sistema y los tiempos de respuesta rápidos en virtud del hecho de que los programas se ejecutan directamente en el controlador de automatización en lugar de en una PC.[13]

Simatic WinCC / SES es muy empleado en las plantas de producción que involucran dosificación, mezcla y transporte de material donde las materias primas almacenadas en tanques, silos o recipientes deben ser combinados en recipientes de reacción y maquinaria de procesamiento en un orden preciso en varias etapas de procesamiento para producir el producto final.

1.5.3 Análisis sobre las recetas en los Sistemas SCADA

Después de haber realizado un breve estudio sobre algunos sistemas SCADA que emplean recetas en la gestión de procesos por lotes se arribó a la conclusión que el empleo de receta en la gestión de procesos por lotes es de vital importancia en aquellas instalaciones para las que dosificación, mezcla y transporte de material son pasos importantes dentro del proceso productivo. El empleo de fórmulas y plantillas permite a los operadores realizar cambios en las misma, sin tener que variar la programación de equipos, ni realizar otra modificación técnica. Al emplear la norma ISA-88 como órgano rector para sus producciones por lotes, se obtiene una mayor flexibilidad de adaptación y soluciones más económicas.

1.6 Metodologías y Tecnologías a utilizar

Las tecnologías y metodologías a utilizar han sido analizadas y especificadas por el equipo de desarrollo del proyecto, debido a esto el desarrollo del presente trabajo sigue las trazadas por el mismo para complementar los estándares establecidos.

1.6.1 Metodología de Desarrollo

Una metodología es aquella guía que se sigue a fin de realizar las acciones propias de una investigación.

En términos más sencillos se trata de la guía que nos va indicando qué hacer y cómo actuar cuando se quiere obtener algún tipo de investigación. Es posible definir una metodología como aquel enfoque que permite observar un problema de una forma total, sistemática y disciplinada.[14]

1.6.1.1 Metodología Extreme Programming(XP)

Extreme Programming es una metodología de desarrollo de la ingeniería de software formulada por Kent Beck en 1999. Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico [15]

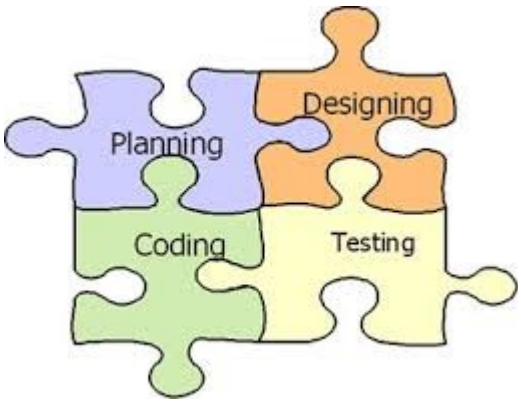

Figura 3: Metodología Extreme Programming.Tomada de la especificación original

Las características fundamentales de XP son:

- **Desarrollo iterativo e incremental**: pequeñas mejoras, unas tras otras.
- **Programación en parejas**: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. La mayor calidad del código escrito de esta manera el código es revisado y discutido mientras se escribe es más importante que la posible pérdida de productividad

inmediata.

- Frecuente **integración del equipo de programación con el cliente** o usuario. Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.
- **Corrección de todos los errores** antes de añadir nueva funcionalidad. Hacer entregas frecuentes.
- **Refactorización del código** reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.
- **Propiedad del código compartida**: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores serán detectados.
- **Simplicidad en el código**: es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

Roles:

- Programador.
- Cliente.
- Encargado de Pruebas (*Tester*).
- Encargado de seguimiento (*Tracker*).
- Entrenador.
- Consultor.

- Gestor.

1.6.2 Lenguaje de Modelado

El Lenguaje Unificado de Modelado (UML) es un lenguaje de modelado visual que se utiliza para especificar, visualizar, construir y documentar artefactos de un sistema de software[16], captura decisiones y conocimientos sobre los sistemas que se deben construir. Se emplea para entender, diseñar, configurar, mantener, y controlar la información sobre tales sistemas. Pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar. Incluye conceptos semánticos, notación, y principios generales. Permite realizar diagramas estáticos, dinámicos, de entorno y organizativos.

1.6.3 Herramientas CASE Visual Paradigm-UML

Las Herramientas CASE (*Computer Aided Software Engineering*), en español Ingeniería de Software Asistida por Computadora, son aplicaciones informáticas utilizadas en el proceso de desarrollo de software en tareas como: realizar un diseño del proyecto, cálculo de costos, implementación de parte del código automáticamente a partir del diseño, compilación automática, documentación o detección de errores entre otras.[17] **Visual Paradigm-UML** es una herramienta CASE profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue.

Las características principales de esta herramienta son las que a continuación enunciamos:

- Soporte para toda la notación UML.
- Capacidades de ingeniería directa (versión profesional) e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de integrarse en los principales Entornos de Desarrollo Integrados (IDE).
- Disponibilidad en múltiples plataformas.

1.6.4 Lenguaje de Programación C++

C++ es un lenguaje de programación diseñado a mediados de los años 1980 por Bjarne Stroustrup. La intención de su creación fue el extender al exitoso lenguaje de programación C con mecanismos que permitan la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, el C++ es un lenguaje híbrido. Posteriormente se añadieron facilidades de programación genérica, que se sumó a los otros dos paradigmas que ya estaban admitidos (programación estructurada y programación orientada a objetos). Por esto se suele decir que el C++ es un lenguaje multiparadigma. Actualmente existe un estándar, denominado ISO C++, al que se han adherido la mayoría de los fabricantes de compiladores más modernos.[18]

Una particularidad del C++ es la posibilidad de redefinir los operadores (sobrecarga de operadores), y de poder crear nuevos tipos que se comporten como tipos fundamentales. C++ permite trabajar tanto a alto como a bajo nivel. A partir de dichas razones y por estudios realizados en el proyecto sobre los diferentes lenguajes de programación posibles a utilizar en el proyecto se seleccionó dicho lenguaje.

1.6.5 Framework Qt

Qt es un *framework* de desarrollo para aplicaciones multiplataforma, que simplifica el desarrollo de aplicaciones en C++ de forma nativa, también puede ser utilizado en otros lenguajes, funciona en las principales plataformas y tiene un amplio apoyo. Es una biblioteca para desarrollar interfaces gráficas de usuario (GUI) y también para el desarrollo de programas sin interfaz gráfica como herramientas de la consola y servidores.

Entre sus componentes podemos encontrar:

- **Las bibliotecas Qt** (clases implementadas en C++).
- ***Qt Designer:*** Para diseñar formularios de forma visual.
- ***Qt Assistant****:* Acceso rápido a la documentación.
- ***Qt Linguist****:* Traducción rápida de programas.
- ***Qmake****:* Simplifica el proceso de construcción de proyectos en las diferentes plataformas

soportadas.

1.6.6 Entorno de Desarrollo Qt Creator

Qt Creator es entorno de desarrollo integrado (**IDE**) multiplataforma, adaptado a las necesidades de los desarrolladores de Qt. *Qt Creator* puede ser ejecutado sobre los sistemas operativos de escritorio Windows, GNU/Linux, Mac OS X, permitiendo a los desarrolladores crear aplicaciones para múltiples escritorios y plataformas de dispositivos móviles.[19]

Características relevantes del IDE *Qt Creator*

- Editor de código sofisticado: El editor de código avanzado de *Qt Creator* ofrece compatibilidad con la edición de los lenguajes C++ y QML (basado en *JavaScript*), ayuda sensible al contexto, finalización de código y mucho más.[19]
- Diseñadores integrados de interfaz de usuario: *Qt Creator* ofrece dos editores visuales integrados, Qt Designer para la creación de interfaces de usuario de *widgets* Qt y *Qt Quick* Designer para el desarrollo de interfaces de usuario animadas con el lenguaje QML. [19]
- Administración de proyectos y versiones: Independientemente de si importas un proyecto existente o creas uno desde cero, *Qt Creator* genera todos los archivos necesarios. Es compatible con *cross-qmake* y *CMake.*
- **Objetivos de ordenadores de escritorio y portátiles:** *Qt Creator* ofrece compatibilidad con la creación y ejecución de aplicaciones Qt para ordenadores de escritorio y dispositivos móviles. La configuración generada te permite cambiar rápidamente entre los destinos de generación.

1.7 Conclusiones Parciales

A partir del estudio realizado sobre el uso de las recetas en distintos sistemas SCADA quedó evidenciada la necesidad de introducir en el SCADA GALBA la gestión de recetas. Además a través del estudio de la Norma ISA 88 se logró definir una clase de receta aplicable al GALBA: la receta básica, la cual en lo adelante permitirá a dicho sistema gestionar procesos por lotes. Por otra parte para el desarrollo del componente se utilizarán las herramientas y tecnologías seleccionadas ya que son las que más se ajustan

para garantizar los objetivos propuestos contribuyendo también a la independencia tecnológica por la que se aboga en la universidad.

Capitulo 2. Análisis y Diseño de la Herramienta

En el presente capítulo se describen las funcionalidades de la solución propuesta, generando además todos los artefactos pertenecientes a cada una de las fases que define la metodología seleccionada. Se presentan los resultados de las fases de exploración y planificación de la solución propuesta. Se especifican además los requisitos funcionales y no funcionales que el sistema debe cumplir para satisfacer las necesidades del cliente, se definen la arquitectura del sistema y los patrones de diseño.

2.1 Propuesta de Solución del Sistema

A partir del marco teórico analizado en el capítulo anterior, se propone el desarrollo de un componente para la gestión y configuración de recetas para el SCADA GALBA. Se implementará un componente que permita la gestión de procesos por lotes mediante el uso de recetas y plantillas de recetas, asociando cada parámetro de fórmula a una variable del sistema. Dicho componente contará con una interfaz mediante la cual el sistema permitirá crear plantillas de recetas y recetas básicas a través de la configuración del sistema y una vez creadas las mismas persistirán en la base de datos de configuración. Dicha interfaz permitirá además navegar, buscar, elegir y ejecutar todas las acciones referentes a la gestión de las recetas.

Una vez descritas las características deseadas para el componente se procede a la identificación de los actores del sistema y al levantamiento de los requisitos primarios, proceso que pertenece al flujo de trabajo de la fase de exploración.

2.2 Fase de Exploración

La metodología XP, comienza en su ciclo de vida con la fase de exploración, proponiendo definir durante esta etapa el alcance general del proyecto; además los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo se realiza una familiarización con las herramientas, tecnologías y prácticas que se emplearán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. Las estimaciones realizadas en esta fase son primarias, ya que están basadas en datos de alto nivel y podrían

variar cuando se analicen con mayor detalle en cada iteración. Para esta fase se sugiere una extensión de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología.[15]

2.2.1 Actores del sistema

Se define como actor(es) del sistema, a aquellas personas que interactúan de una forma u otra con los mecanismos propuesto, demandando una o más funcionalidades del mismo, que pueden estar vinculadas al proceso de desarrollo o no. Esto incluye tanto a los operadores humanos como a sistemas externos. En el caso particular de la gestión de recetas en el GALBA los actores del sistema serían un operador dedicado al funcionamiento del SCADA, que como actor del sistema llamaremos **operador de planta** y un mantenedor del sistema que como su nombre lo indica será el encargado de mantener el sistema que como actor del sistema lo llamaremos **administrador**.

Actores del Sistema	**Descripción**
Operador de planta	Es la persona encargada de realizar la interacción con el sistema para la ejecución de las funcionalidades: crear, modificar y activar receta.
Administrador	Es la persona encargada de mantener el sistema, además de la ejecución de las funcionalidades: crear, modificar y eliminar una plantilla de receta, así como eliminar una receta. Un administrador posee privilegios para la ejecución de las funcionalidades propias de un operador de planta.

Tabla 1: Actores del sistema

2.2.2 Requisitos funcionales del sistema

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir. El propósito fundamental de la captura de requisitos es guiar el desarrollo hacia un sistema correcto. De acuerdo con los objetivos propuestos el sistema debe ser capaz de:

Código	Asignado a	Descripción	Prioridad	Estimación
		Gestionar parámetro de fórmula		7 días
Rf # 1	Yosvany	Crear parámetro de fórmula	Muy Alta	2 días
Rf # 2	Yosvany	Modificar parámetro de fórmula	Muy Alta	2 días
Rf # 3	Yosvany	Mostrar parámetro de fórmula	Muy Alta	2 días
Rf # 4	Yosvany	Eliminar parámetro de fórmula	Muy Alta	1 día
		Gestionar plantilla de receta		7 días
Rf # 5	Yosvany	Crear plantilla de receta	Muy Alta	2 días
Rf # 6	Yosvany	Modificar plantilla de receta	Muy Alta	2 días
Rf # 7	Yosvany	Mostrar plantilla de receta	Muy Alta	2 días
Rf # 8	Yosvany	Eliminar plantilla de receta	Muy Alta	1 día
		Gestionar receta		7 días
Rf # 9	Yosvany	Crear receta	Muy Alta	2 días
Rf # 10	Yosvany	Modificar receta	Muy Alta	2 días
Rf # 11	Yosvany	Mostrar receta	Muy Alta	2 días
Rf # 12	Yosvany	Eliminar receta	Muy Alta	1 día
Rf # 13	Yosvany	Activar receta	Muy Alta	7 días
Rf # 14	Yosvany	Registro de acciones de usuario	Alta	4 días

Tabla 2: Requisitos funcionales

2.2.3 Requisitos no funcionales del sistema

En el siguiente epígrafe se detallan los requisitos no funcionales que se tuvieron en cuenta para el diseño y desarrollo de los prototipos del sistema. Los requisitos no funcionales son propiedades o cualidades que

debe cumplir el producto. Debe pensarse en ellos como las características que hacen al producto atractivo, usable, rápido y confiable.

Código	**Clasificación**
	Usabilidad
Rnf # 1	El usuario será informado de manera detallada en el momento que se inserten los datos de forma correcta, así como cuando ocurra un error en la entrada de los mismos.
	Confiabilidad
Rnf # 2	El sistema deberá chequear antes de realizarse las configuraciones o solicitudes, que el usuario en cuestión tenga los permisos requeridos. Al intentar realizar una configuración se realizará un chequeo de privilegios del usuario y solo si presenta permisos de configuración podrá realizar las modificaciones, de lo contrario debe enviarse un mensaje al usuario donde se explica el motivo del rechazo.
	Hardware
Rnf # 3	Para garantizar un buen desempeño del componente, las PC clientes deben poseer 1GB de memoria RAM como mínimo, un procesador Pentium 4 o superior y 10 GB de almacenamiento como mínimo.
	Software
Rnf # 4	Sistema Operativo: GNU/Linux – Debian 7(Wheezy)
Rnf # 5	*Framework* Qt versión 4.8
Rnf # 6	Para la sincronización de hilo *boots-thread versión 1.49*
Rnf # 7	Para la serialización de los datos *boots-serialization versión 1.49*

Tabla 3: Requisitos no funcionales

2.2.4 Historias de Usuarios

Las historias de usuario son la técnica utilizada en XP para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible, en cualquier momento las historias de usuario pueden romperse, reemplazarse por otras más específicas o generales, añadirse nuevas o ser modificadas. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los desarrolladores puedan implementarla en unas semanas.[20]. Los desarrolladores dialogarán de forma directa con los clientes cuando llegue la hora de la implementación, para así obtener los detalles necesarios.

Los contenidos de las fichas de las HU[3] quedan estructurados de la siguiente manera:

Número: A cada HU se le asigna un número para facilitar su identificación por parte del equipo de desarrollo.

Nombre de HU: Nombre descriptivo de la HU.

Fecha: Fecha en la cuál fue redactada la HU.

Usuario: Actor protagonista de la HU.

Prioridad en el Negocio: Grado de prioridad que le asigna el cliente a la HU en dependencia de sus necesidades (Alta, Media o Baja).

Riesgo en Desarrollo: Grado de complejidad que le asigna el equipo de desarrollo a la HU luego de analizarla (Alto, Medio o Bajo).

Tipo de Actividad: Define el tipo de actividad en la HU (nueva, corrección, mejora).

Iteración Asignada: Número de la Iteración en la cual será implantada la HU.

Puntos Estimados: Unidades de tiempo estimadas por el equipo de desarrollo para darle cumplimiento a la HU. Una unidad de tiempo equivale a una semana de trabajo de 40 horas.

Puntos Reales: Unidades de tiempo reales que el equipo de desarrollo necesitó para darle cumplimiento

3 En lo adelante se empleará estas siglas para referirnos a las Historias de Usuarios

a la HU. Una unidad de tiempo equivale a una semana de trabajo de 40 horas.

Descripción: Descripción simple sobre lo que debe hacer la funcionalidad a la que se hace referencia.

Observaciones: Aquellos detalles relevantes que serán resueltos tras la conversación del equipo desarrollador con el cliente.

A continuación se muestran las Historias de Usuario definidas por el equipo de desarrollo las restantes se encuentran en los anexos:.

Historia de Usuario	
Número: 1	**Nombre de HU:** Gestionar Parámetro de Fórmula
Fecha: 10-02-2015	**Usuario:** Administrador
Prioridad en el Negocio: Alta	Riesgo de Desarrollo: Medio
Tipo de actividad: nueva	**Iteración Asignada:**1
Puntos Estimados:1	Puntos Reales: 2
Descripción:El sistema debe ser capaz de crear, modificar, mostrar y eliminar parámetros de fórmula. Además el sistema debe ser capaz de adicionar parámetros a una plantilla de receta, cada parámetro estará asociado a una variable del sistema.	
Observaciones: Un parámetro de fórmula se refiere a un material o condición necesario para la obtención de una receta. Este deberá estar directamente enlazado con una variable de algún módulo de adquisición del proyecto.	

Tabla 4: Gestionar Parámetro de Fórmula

Historia de Usuario	
Número: 5	**Nombre de HU**: Registrar acciones sobre recetas
Fecha: 10-02-2015	**Usuario:** Operador de planta.

Prioridad en el Negocio: Alta	**Riesgo de Desarrollo:** Medio
Tipo de actividad: nueva	**Iteración Asignada:** 3
Puntos Estimados: 1	**Puntos Reales**: 1
Descripción: El sistema deberá registrar cada acción sobre las recetas. Por cada evento de creación, modificación, eliminado y activación de una receta se enviará hacia históricos un log que contenga el ID de la receta y el operador que realizó la acción.	
Observaciones:	

Tabla 5: Gestionar Plantilla

2.3 Fase de Planificación

La planificación es una fase corta donde el cliente establece la prioridad de cada HU, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Típicamente esta fase consiste en una o varias reuniones grupales de planificación y el resultado es un Plan de Entregas. Esta fase dura unos pocos días. Las estimaciones de esfuerzo asociadas a la implementación de las historias la establecen los programadores utilizando como medida el punto, lo que equivale a una semana ideal[4] de programación. Las historias generalmente valen de 1 a 3 puntos. La planificación se puede realizar basándose en el tiempo o el alcance. Al planificar por tiempo, se multiplica el número de iteraciones por la velocidad del proyecto, determinándose cuantos puntos se pueden completar. Al planificar según el alcance, se divide la suma de puntos de las historias de usuario seleccionadas entre la velocidad del proyecto, obteniendo el número de iteraciones necesarias para su implementación.[15]

2.3.1 Estimación de esfuerzos por historia de usuario

Teniendo en cuenta la prioridad que tiene una determinada HU en el desarrollo del componente se decide en que iteración será implementada. Las HU que cuentan con mayor importancia por ser funcionalidades indispensables para el funcionamiento del componente, deben ser implementadas en las primeras iteraciones del ciclo de desarrollo.

4 Trabajando 40 horas a la semana.

A continuación se muestra mediante una tabla la planificación de las diferentes HU para cada iteración teniendo en cuenta su prioridad.

No	Historia de Usuario	Prioridad	Esfuerzo Estimado
1	Gestionar Parámetro de Fórmula	Alta	0,9 puntos
2	Gestionar Plantilla	Alta	1,4 puntos
3	Gestionar Receta	Alta	1,6 puntos
4	Activar Receta	Alta	1 puntos
5	Registro de Acciones sobre Recetas	Alta	0,6 puntos

Tabla 6: Estimación de esfuerzos

2.3.2 Plan de Iteraciones

Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. El Plan de entrega está compuesto por iteraciones de no más de 3 semanas de duración. En la primera iteración se intenta establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto; esto se logra estableciendo las historias de usuarios que fuercen la creación de la arquitectura antes mencionada, sin embargo esto no siempre es posible ya que es el cliente quien decide que historias se implementarán en cada iteración (para maximizar el valor de negocio). Al final de la última iteración el sistema estará listo para entrar en producción.[20]

Una vez definidas las Historias de Usuarios y estimado el esfuerzo propuesto para la realización de cada una de ellas, se decidió por el equipo de desarrollo realizar el sistema en 3 iteraciones, las cuales se describen de manera más detallada a continuación:

Iteración 1

Esta iteración tiene como objetivo darle cumplimiento a las HU que se consideraron con mayor importancia para el desarrollo de la herramienta. Al concluir dicha iteración el sistema contará con todas las funcionalidades descritas en las HU 1 y la HU 2, las cuales aluden a la creación, modificación y eliminado de un parámetro de fórmula, así como de una plantilla de receta, además de la asociación de

cada parámetro a una variable del sistema.

Iteración 2

Esta iteración tiene como objetivo darle cumplimiento a la HU 3, la cual responde a la creación, modificación, eliminado de una Receta, las receta se crearán a partir de plantillas previamente definidas.

Iteración 3

Esta iteración tiene como finalidad darle cumplimiento a la HU 4 la cual se encarga de la activación de una receta, que no es más que escribir los valores que componen a la receta hacia las variables del sistema asociadas a cada parámetros. Además da cumplimiento a la HU 5 la cual responde al registro de todas y cada una de las acciones que se realicen sobre las recetas.

2.3.3 Plan de Duración de las Iteraciones

Como parte del ciclo de vida de un proyecto guiado por la metodología de desarrollo XP, se crea el plan de duración de las iteraciones que se llevarán a cabo durante el desarrollo del proyecto. Este plan tiene como objetivo fundamental mostrar la duración de cada una de las iteraciones en las que está dividida la fase de desarrollo del proyecto, así como el orden en que serán implementadas las HU en cada iteración según la prioridad asignada por el cliente.

Iteración	Historias de Usuarios	Duración total de la iteración
Iteración 1	Gestionar Parámetro de Fórmula Gestionar Plantilla	2,3 semanas 92 horas
Iteración 2	Gestionar Receta	1,6 semanas 64 horas
Iteración 3	Activar Receta Registro de acciones sobre Recetas	1,6 semanas 64 horas

Tabla 7: Plan de duración de las iteraciones

2.4 Diseño de la Solución Propuesta

La metodología X.P sugiere que hay que conseguir diseños simples y sencillos. Hay que procurar hacerlo todo lo menos complicado posible para conseguir un diseño fácilmente entendible e implementable que a la larga costará menos tiempo y esfuerzo desarrollar. Dicha metodología, no requiere la descripción del sistema mediante diagramas de clase utilizando la notación UML, sino que en su lugar se guía por técnicas como las tarjetas CRC (Contenido, Responsabilidad y Colaboración). Esto no implica que no se empleen dichos diagramas para obtener una mejor visión y comunicación entre el equipo de trabajo, siempre y cuando no posea una alta complejidad y defina información importante.

2.4.1 Tarjetas CRC

El uso de las tarjetas CRC permiten al programador centrarse y apreciar el desarrollo orientado a objetos olvidándose de los malos hábitos de la programación procedural clásica. Una tarjeta CRC no es más que una ficha de papel o cartón que representa a una entidad del sistema a las cuales asigna responsabilidades y colaboraciones.[21] Las tarjetas CRC representan objetos; la clase a la que pertenece el objeto se puede escribir en la parte de arriba de la tarjeta, en una columna a la izquierda se pueden escribir las responsabilidades u objetivos que debe cumplir el objeto y a la derecha, las clases que colaboran con cada responsabilidad. La mayor ventaja de las tarjetas CRC es permitir reducir el modo de pensar procedural y apreciar la tecnología de objetos. Las tarjetas CRC permiten contribuir al diseño a todo el equipo del proyecto.[20] A continuación se definen algunas de las tarjetas CRC del sistema, las restantes se encuentran en los anexos:

Basic_Recipe	
Súper Clase:	
Responsabilidad	**Colaboración**
Crear una receta básica Modificar una receta básica Mostrar una receta básica Eliminar una receta básica	Class Encabezado Class Parameter

Tabla 8: Tarjeta CRC Basic_Recipe

Encabezado	
Súper Clase:	
Responsabilidad	**Colaboración**
Crear un encabezado para una receta. Modificar un encabezado para una receta. Mostrar un encabezado para una receta. Eliminar un encabezado para una receta.	

Tabla 9: Tarjeta CRC Encabezado

Parameter	
Súper Clase:	
Responsabilidad	**Colaboración**
Crear un parámetro de fórmula Modificar un parámetro de fórmula	

Mostrar un parámetro de fórmula Eliminar un parámetro de fórmula	

Tabla 10: Tarjeta CRC Parameter

2.4.2 Diagrama de paquetes

Los diagramas de paquetes tienen como objetivo obtener una visión más clara del sistema de información orientado a objetos, organizándolo en subsistemas, agrupando los elementos del análisis, diseño o construcción y detallando las relaciones de dependencia entre ellos [22].

El siguiente esquema muestra los distintos paquetes que componen el Planificador de tareas, sobre los cuales serán insertadas las clases y métodos necesarios para el gestión y configuración de recetas los cuales se explican a continuación:

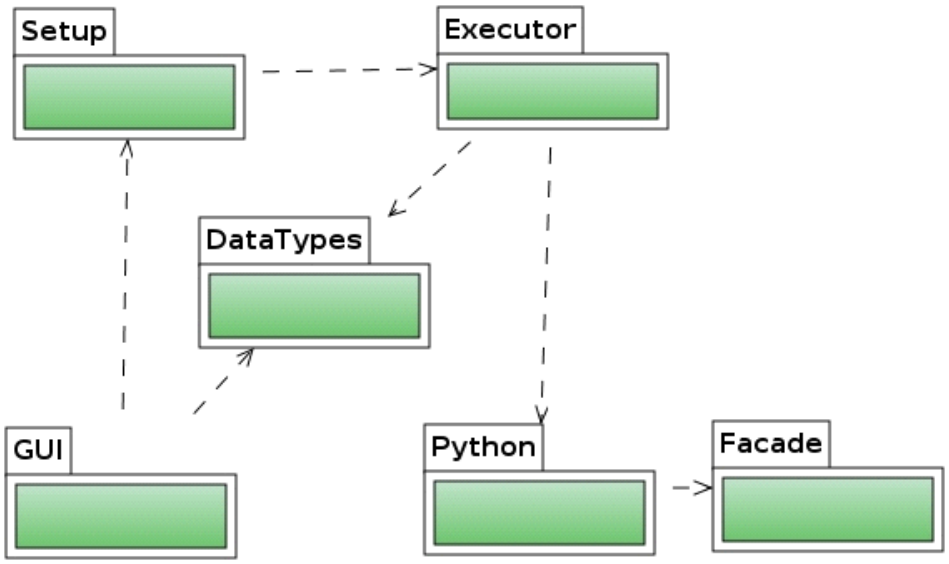

Figura 4: Diagrama de paquetes del Planificador de Tareas del GALBA

- **Executor:**

En este paquete se encuentran las clases que definen la lógica para la configuración y gestión de las plantillas de recetas, los parámetros de fórmula así como las recetas además es el encargado de la

ejecución de las recetas.

- **GUI:**

En este paquete se encuentran las clases que definen las interfaces gráficas para la gestión de las recetas, las plantillas de recetas y los parámetros de fórmula.

- **Setup:**

Este paquete contendrá la lógica para mantener la persistencia de los objetos pues es el que permitirá cargar una configuración y realizar las modificaciones en caliente mediante el módulo de configuración del **SCADA GALBA**, además contiene los tipos de datos a configurar.[23]

- **Facade:**

  Este paquete contendrá la lógica para la comunicación con los distintos módulos del SCADA GALBA.

- **DataTypes**

Este paquete contiene todas las entidades del módulo Aplicaciones, como son los tipos de tareas, los eventos, así como las plantillas de recetas, el encabezado y los parámetros que definen la fórmula simplificada de la receta.

2.5 Arquitectura del Sistema

El diseño de la arquitectura de un sistema, es el proceso mediante el cual se define una solución para los requisitos técnicos y operacionales del mismo. Este proceso define qué componentes conforman el sistema, cómo se relacionan entre ellos, y como mediante su interacción llevan a cabo las funcionalidades especificadas.

Para el diseño de la herramienta se seleccionó una arquitectura N capas, específicamente 3 capas, que es la arquitectura que sigue el módulo Aplicaciones. Esta arquitectura define 3 capa fundamentales: la capa de presentación, la capa de lógica del negocio y la capa de acceso a datos. La arquitectura por capa es un estilo cuyo objetivo primordial es la separación entre la lógica de negocio y la lógica de diseño. Esta se seleccionó por su capacidad de permitir tener acopladas las clases del componente y conocer el flujo

de datos generado por las mismas.

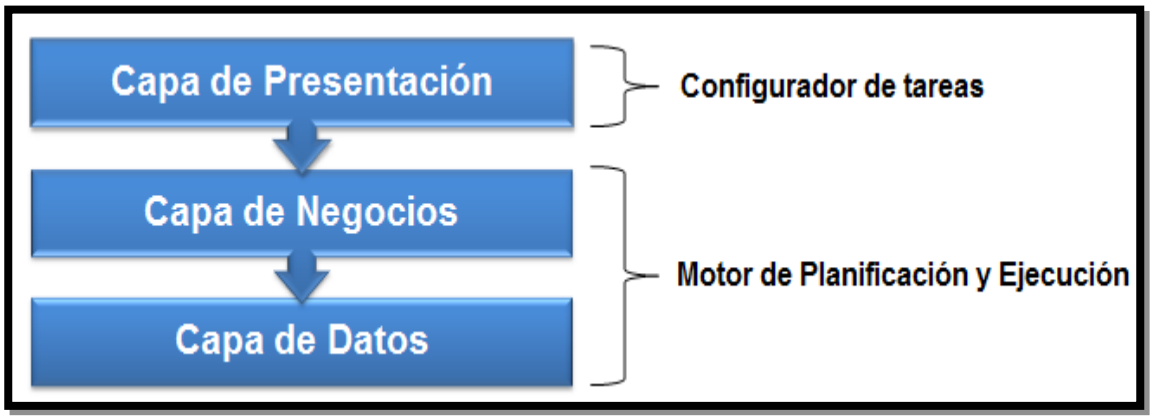

Figura 5: Arquitectura del módulo Aplicaciones

2.5.1 Patrones de diseño

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software. En otras palabras, brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares. Debemos tener presente los siguientes elementos de un patrón: su nombre, el problema (cuando aplicar un patrón), la solución (descripción abstracta del problema) y las consecuencias (costos y beneficios).[24]

En el desarrollo de la solución se emplearon los siguientes patrones **GRASP[5]:**

- **Experto:** Asigna una responsabilidad a la clase que cuenta con la información necesaria para cumplirla. Esto nos indica que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo obtendremos un diseño con mayor cohesión y así la información se mantiene encapsulada (disminución del acoplamiento).
- **Creador:** Guía la asignación de responsabilidades relacionadas con la creación de objetos. Se aplica en todos los casos donde una clase tiene la responsabilidad de crear una nueva instancia de la otra. El patrón creador ayuda a identificar quién debe ser el responsable de la creación (o instanciación) de nuevos objetos o clases. La nueva instancia deberá ser creada por la clase que:

5 General Responsibility Assignment Software Patterns

- Tiene la información necesaria para realizar la creación del objeto.
- Usa directamente las instancias creadas del objeto.
- Almacena o maneja varias instancias de la clase.

- **Bajo Acoplamiento:** El bajo acoplamiento fomenta el aumento de la reutilización y la eliminación de las redundancias, creando clases más independientes y con mayor resistencia al impacto de los cambios, que aumentan la productividad y la posibilidad de reutilización. Este patrón se basa en la idea de tener las clases lo menos ligadas entre sí. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases.
- **Alta Cohesión:** La cohesión es una medida de cuan relacionadas y enfocadas están las responsabilidades de una clase, además de que una alta cohesión garantiza que clases con responsabilidades estrechamente relacionadas no realicen un trabajo enorme, y que cada elemento de nuestro diseño debe realizar una labor única dentro del sistema.

  En la solución este patrón es el resultado de asignar responsabilidades a cada uno de los componente, por ejemplo cada una de las clases (controladoras) en el sistema realizan una sola labor no desempeñada por el resto.

  Los patrones **GOF**[6] empleados en el diseño de la solución son:

- **Abstrac Factory (Fábrica Abstracta):** Se utiliza este patrón al trabajar con objetos de distintas familias de manera que no se mezclen entre sí, haciendo transparente el tipo de familia concreta que se esté usando. Cuando el marco de trabajo necesita, por ejemplo crear un nuevo objeto, busca en la definición de la factoría el nombre de la clase que se debe utilizar para esta tarea.
- **Singlenton (Instancia Única):** Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. En el desarrollo de la se

6 Gang of Four

evidenciará el empleo de este patrón en las clases *ExecutorRecipe, ExecutorManagerTemplate* para garantizar la existencia de una única instancia de cada una ya que las mismas son la encargadas ejecutar las recetas una y la otra del controlar las plantillas de recetas.

- **Observer (Observador):** Este patrón define una relación de dependencia del tipo uno a muchos entre objetos de manera que cuando uno de los objetos cambia su estado, notifica este cambio a todos los dependientes. Este patrón por lo general es bastante flexible y sencillo a la vez. Se utiliza cuando uno quiere notificar a otros objetos de un evento. En principio, lo que sucede es que un Objeto (llamémoslo *Observador*) se inscribe a otro Objeto (llamémoslo *Sujeto*) y este le avisa cuando un evento es disparado (o cuando el estado del *Sujeto* ha cambiado).

  En la solución este patrón se observa entre la clase Basic_Recipe y la *RecipeTemplate* ya que la clase Basic_Recipe está obligada a avisar a la clase *RecipeTemplate* cuando una instancia de la misma ha sido eliminada o creada.

2.6 Estándar de codificación

Los estándares de codificación, también conocidos como estilos de programación o convenciones de código, son convenios para escribir códigos fuentes en ciertos lenguajes de programación. Permiten que el código en consecuencia sea mantenible y que todos los participantes lo pueden entender en un menor tiempo.[25]

Para la implementación del componente en cuestión se empleó el Estándar de codificación de C++ para el proyecto SCADA GALBA.[26]

Algunas de las pautas que define el estándar utilizado define:

- En los archivos cabecera debe incluir el *copyright* y la licencia, o una referencia de la misma, al estilo GNU[7] GPL[8].
- Se adopta el estilo de bloques de documentación de JavaDoc, el cual consiste de un bloque de comentario de estilo C.

7 Acrónimo recursivo “GNU is Not UNIX”
8 Licencia Pública General (en inglés General Public License)

- Para hacer una descripción breve se adopta el uso del comando @brief.
- Es importante especificar el nombre del autor y la fecha de creación de cualquier estructura en un código, para ello se utilizan los comandos @autor y @date.
- Para hacer referencia a otras clases utilizar el comando @see.
- El código será escrito en inglés y la documentación en español.
- Las variables y funciones comienzan con letra minúscula. Cada palabra consecutiva en el nombre comienza con letra mayúscula.

2.7 Conclusiones Parciales

La elaboración de las Historias de Usuarios como descripción de las funcionalidades y la aplicación de los patrones de diseño y arquitectónicos que se ajustaron al desarrollo facilitaron el diseño e implementación del componente permitiendo al equipo de desarrollo obtener una visión mas clara del mismo. Se concluye además, que todos los artefactos generados por la metodología en esta etapa guiarán de forma efectiva el desarrollo de la componente, o prácticamente cualquier otro subsistema que pueda ser concebido en el futuro.

Capitulo 3. *Implementación y Pruebas*

En el presente capítulo se detallarán las tres iteraciones llevadas a cabo durante la etapa de construcción del sistema, definiendo las tareas generadas a partir del desarrollo de las historias de usuario durante las iteraciones planificadas. Además quedarán especificados los resultados obtenidos de la ejecución de las pruebas de aceptación previamente diseñadas para probar las funcionalidades descritas y la valoración de dichas pruebas de manera cualitativa.

3.1 Desarrollo de las Iteraciones

En la fase de Planificación se detallaron las historias de usuario correspondientes a cada una de las iteraciones para desarrollar el sistema, teniendo presente las necesidades requeridas por el cliente. Durante el transcurso de cada iteración se realizó una revisión del plan de iteraciones. Como parte de este plan se desglosaron las historias de usuario definidas en tareas de programación o ingeniería, asignándole a un equipo de desarrollo o a una persona la responsabilidad de su implementación. Estas tareas son para el uso estricto del desarrollador-programador, por lo que no fue necesario su descripción en un lenguaje no técnico para hacerlas entendibles al cliente.

A continuación se especifica con mayor detalle las tareas de desarrollo que se realizaron en cada una de las iteraciones:

Iteración 1

Esta iteración tuvo como objetivo dar cumplimiento a las HU que se consideraron con una mayor importancia para el desarrollo del componente. Al concluir dicha iteración se contó con todas las funcionalidades descritas en las HU1 y HU2, la primera se refiere a la creación, modificación y eliminación de los parámetros de fórmula, así como enlazando cada parámetro de fórmula a una variable del SCADA. La HU2 gestionar plantillas de recetas referente a la creación, modificación y eliminación de plantillas de recetas.

Historias de Usuario	Tiempo de implementación(semanas)	
	Estimación	**Real**
Gestionar parámetros de fórmula	1,3 semanas	1,4 semanas
Gestionar plantillas de recetas	1 semana	1 semana
Total	2,3 semanas	2,4 semanas

Tabla 11: Tiempo de estimación para la primera iteración

A continuación se muestran algunas de las tareas de ingeniería llevadas a cabo durante el transcurso de la iteración, las tareas restantes se encuentran en los anexos.

Tarea de Ingeniería	
No. de tarea: 1	**No. de HU:**1
Nombre de tarea: Diseñar interfaz de una extensión para el planificador de tareas que permita la gestión de parámetros.	
Tipo de tarea: Diseño	**Puntos estimados**: 0,2
Fecha inicio: 11:03:2015	**Fecha fin** 11:03:2015
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Tiene como fin, crear la interfaz de la herramienta que se integrará al planificador de tareas para la gestión de la lógica operacional.	

Tabla 12: Tarea de ingeniería: "Diseñar interfaz para parámetros"

Tarea de Ingeniería	
No. de tarea: 2	**No. de HU:**1
Nombre de tarea: Crear parámetro de fórmula	
Tipo de tarea: Implementación	**Puntos estimados:** 0,2
Fecha inicio: 12:03:2015	**Fecha fin** 12:03:2015
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Funcionalidad encargada de crear los parámetros de fórmula, defiendo los atributos de los mismos y enlazando el parámetro con la variable del SCADA correspondiente.	

Tabla 13: Tarea de ingeniería: "Crear parámetro de fórmula"

Tarea de Ingeniería	
No. de tarea: 3	**No. de HU:**1
Nombre de tarea: Modificar parámetro de fórmula	
Tipo de tarea: Implementación	Puntos estimados: 0,4
Fecha inicio: 12:03:2015	Fecha fin: 13:03:2015
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Funcionalidad encargada de modificar los parámetros de fórmula asociados a un plantilla de receta o una receta.	

Tabla 14: Tarea de ingeniería: "Modificar parámetro de fórmula"

Iteración 2

Esta iteración tuvo como objetivo darle cumplimiento a la HU2 la cual responde al proceso de la gestión de recetas, durante el transcurso de la misma se diseñó la interfaz visual que será incluida al planificador de tareas para la gestión de las recetas. Al concluir dicha iteración el sistema contará con todas las funcionalidades necesarios la para gestión de una receta.

Historias de Usuario	Tiempo de implementación(semanas)	
	Estimación	**Real**
Gestionar recetas	1, 6 semanas	1,5 semanas
Total	1,6 semanas	1,5 semanas

Tabla 15: Tiempo de estimación para la segunda iteración

A continuación se muestran algunas de las tareas de ingeniería llevadas a cabo durante el transcurso de la iteración, las tareas restantes se encuentran en los anexos.

Tarea de Ingeniería	
No. de tarea: 10	**No. de HU:** 3
Nombre de tarea: Diseñar interfaz de una extensión para el planificador de tareas que permita la gestión de recetas.	
Tipo de tarea: Diseño	**Puntos estimados:** 0,2
Fecha inicio: 23:03:2015	**Fecha fin:** 23:03:2015
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Tiene como fin, crear la interfaz de la herramienta que se integrará al planificador de tareas para la gestión de la lógica operacional.	

Tabla 16: Tarea de ingeniería: "Diseñar interfaz para recetas"

Tarea de Ingeniería	
No. de tarea: 11	**No. de HU:** 3
Nombre de tarea: Crear receta	
Tipo de tarea: Implementación	**Puntos estimados:** 0,4

Fecha inicio: 24:03:2015	**Fecha fin**: 25:03:2015
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Funcionalidad encargada de crear las plantilla de receta, definiendo todos los atributos de las recetas y asignando los valores correspondiente a los parámetros de la fórmula de la receta.	

Tabla 17: Tarea de ingeniería: "Crear receta"

Tarea de Ingeniería	
No. de tarea: 12	**No. de HU:** 3
Nombre de tarea: Modificar receta.	
Tipo de tarea: Implementación	**Puntos estimados:** 0,2
Fecha inicio: 26:03:2015	**Fecha fin:** 26:03:2015
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Funcionalidad encargada de modificar las recetas.	

Tabla 18: Tarea de ingeniería: "Modificar receta"

Tarea de Ingeniería	
No. de tarea: 13	**No. de HU:** 3
Nombre de tarea: Eliminar recetas	
Tipo de tarea: Implementación	**Puntos estimados:** 0,2
Fecha inicio: 28:03:2015	**Fecha fin**: 28:03:2015
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Funcionalidad encargada de eliminar las recetas existentes, siempre y cuando las mismas no se encuentren en ejecución.	

Tabla 19: Tarea de Ingeniería "Eliminar receta"

Iteración 3

El objetivo de la presente iteración es darle cumplimiento a las HU3 y HU4, las que describen los procesos de activación de una receta y el registro de las acciones de usuarios. La primera encargada de escribir sobre las variables del SCADA y la segunda de llevar un registro de logs de las principales acciones del usuario sobre las recetas y las plantillas de recetas.

Historias de Usuarios	**Tiempo de implementación(semanas)**	
	Estimación	**Real**
Activar receta	1 semana	0,9 semana
Registrar acciones de usuario	0,6 semanas	0,8 semana
Total	1,6 semanas	1,7 semanas

Tabla 20: Tiempo de estimación para la tercera iteración

A continuación se muestran algunas de tareas de ingeniería llevadas a cabo durante el transcurso de la iteración, las tareas restantes se encuentran en los anexos.

Tarea de Ingeniería	
No. de tarea: 14	**No. de HU:** 4
Nombre de tarea: Activar receta	
Tipo de tarea: Implementación	Puntos estimados: 1
Fecha inicio: 02:04:2015	Fecha fin: 08:04:2015
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Funcionalidad encargada de realizar la escritura de los valores de cada parámetro hacia las variables del sistema a la que están asociados.	

Tabla 21: Tarea de ingeniería: "Activar receta"

Tarea de Ingeniería	
No. de tarea: 15	**No. de HU:** 4
Nombre de tarea: Registro de acciones de usuario.	
Tipo de tarea: Implementación	Puntos estimados:
Fecha inicio: 09:04:2015	Fecha fin: 13:04:2015
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Funcionalidad encargada de plasmar las acciones realizadas por el usuario en el sistema.	

Tabla 22: Tarea de ingeniería: "Registro de acciones de usuario"

3.2 Pruebas

La metodología XP define como uno de sus pilares fundamentales el proceso de pruebas, siendo este un proceso vital para garantizar la calidad del producto desarrollado. Esta metodología anima a probar tanto como le sea posible al equipo de desarrollo permitiéndole reducir el número de errores no detectados y disminuir el tiempo entre la aparición de un error y su detección. En este caso también permitió aumentar la seguridad al evitar efectos colaterales no deseados realizando modificaciones y refactorizaciones. XP divide las pruebas en dos grupos: pruebas unitarias, encargadas de verificar el código y diseñadas por los programadores y las pruebas de aceptación o pruebas funcionales destinadas a evaluar si al final de cada iteración se consiguió la funcionalidad requerida por el cliente.[27]

3.2.1 Pruebas de aceptación.

Las pruebas de aceptación tienen mayor importancia que las pruebas unitarias, dado que significan la satisfacción del cliente con el producto desarrollado, el final de una iteración y el comienzo de la siguiente, por consecuente el cliente es la persona indicada para diseñar las pruebas a ejecutar.[27] La ejecución de las pruebas previamente diseñadas, permitió la evaluación de las funcionalidades de la solución desarrollada antes de implantarlo en su entorno real de explotación. Algunos de los resultados obtenidos al aplicar de dichas pruebas se muestran a continuación, el resto se encuentra en los anexos.

<table>
<tr><th colspan="3">Caso de Prueba de Aceptación</th></tr>
<tr><td>Número: 1</td><td colspan="2">Historia de usuario: Crear parámetro de fórmula.</td></tr>
<tr><td colspan="3">Probador encargado: Yosvany Leyva Pizarrosa</td></tr>
<tr><td colspan="3">Descripción: Se comprueba la funcionalidad crear parámetro de fórmula a través de la interfaz visual.</td></tr>
<tr><td colspan="3">Condiciones de ejecución:
1. Debe existir una configuración con recursos de tipo puntos (analógicos o digitales) de entrada o salida disponibles para asignar a cada parámetro de fórmula a crear.</td></tr>
<tr><td colspan="3">Entrada/Pasos de ejecución:
1. El usuario selecciona la opción crear nuevo parámetro, que aparece en el menú principal de la y el acceso directo de la barra de herramientas.
2. El usuario selecciona la variable del SCADA asociada al parámetro.
3. El usuario selecciona botón “Aceptar”.</td></tr>
<tr><th>Casos válidos</th><th>Resultado esperado</th><th>Resultado de prueba</th></tr>
<tr><td>Insertar nombre del parámetro ,
insertar descripción del mismo, seleccionar la unidad de medida del parámetro y seleccionar el tipo de datos en que se encuentra el valor. Debe tenerse en cuenta que todos los campos son obligatorios.</td><td>Luego de adicionar el parámetro, si el proceso ha resultado correcto se actualiza la base de datos con el nuevo parámetro.</td><td>Satisfactorio</td></tr>
<tr><th>Casos no válidos</th><th>Resultado esperado</th><th>Resultado de prueba</th></tr>
</table>

El parámetro se encuentra insertado en el sistema. No asociar recurso del SCADA. Campos requeridos vacíos.	Se debe mostrar un mensaje de error indicando que el parámetro se encuentra insertado en el sistema o que existen campos obligatorios sin llenar.	Satisfactorio
Evaluación de la prueba: Satisfactorio		

Tabla 23: Caso de prueba "Crear parámetro de fórmula"

Caso de Prueba de Aceptación		
Número: 8	**Historia de usuario:** Modificar receta.	
Probador encargado: Yosvany Leyva Pizarrosa		
Descripción: Permitir que el usuario modifique una recetas a través de la interfaz visual.		
Condiciones de ejecución: 1. El usuario que interactúa con el sistema debe tener privilegios para ejecutar la operación modificar recetas (operador o administrador).		
Entrada/Pasos de ejecución: 1. El usuario selecciona la receta a modificar accionando doble clic sobre la misma. 2. El usuario selecciona botón “Aceptar”.		
Casos Válidos	**Resultado esperado**	**Resultado de prueba**
Insertar los datos que desea modificar (nombre, versión o descripción) o seleccionar el nuevo estado de la receta. El usuario no pude dejar campos vacíos.	Luego de modificar los campos estimados por el usuario, si el proceso ha resultado correcto se actualizan la base de datos y la vista con los nuevos datos de	Satisfactorio.

	la receta.	
Casos no válidos	**Resultado esperado**	**Resultado de prueba**
Sustituir el nombre de la receta por otro nombre igual al de una receta ya existente en el sistema.	Se debe mostrar un error indicando que existe una receta con igual nombre.	Satisfactorio.
Evaluación de la prueba: Satisfactorio		

Tabla 24: Caso de prueba "Modificar receta"

3.2.2 Resultados esperados

Para llevar a cabo las pruebas de aceptación se realizaron 3 iteraciones en las cuales se detectaron varios tipos de errores, siendo los siguientes los más significativos:

1. Campos de entrada de datos sin validar, entrada de caracteres extraños por ejemplo: @,#,$, %,&,*,entre otros.
2. Faltas de ortografía en los mensajes mostrados al usuario.
3. Algunos botones “Eliminar” no funcionaban correctamente.
4. El botón “Activar Receta” no funcionaba correctamente.

La siguiente tabla detalla el proceso de prueba y los resultados obtenidos.

Sistema	HU	Iteración	NC[9]	Cerrada(s)	No Procede
Componente para gestión de procesos por lotes.	5	1ra	6	6	0
		2da	2	2	0
		3ra	0	0	0

Tabla 25: Plan de resultado

9 No conformidades detectadas durante las iteraciones.

3.2.3 Análisis de los Resultados

Se realizaron las pruebas de aceptación las cuales demostraron el buen funcionamiento del sistema al lograr la gestión de recetas, las plantillas de recetas, además de los parámetros de fórmula. Se comprobó el correcto funcionamiento de la funcionalidad que permite enlazar un parámetro de fórmula con una variable de campo del SCADA. Luego de comprobadas cada una de las funcionalidades correspondientes a la gestión de recetas se procedió a comprobar que una vez activada la receta los valores correspondiente a su fórmula simplificada sean escritos en las variables de campo asociadas a los mismos demostrando de este modo que el componente desarrollado da cumplimiento a la problemática planteada al inicio del presente trabajo.

3.3 Conclusiones Parciales

La elaboración de las tareas de programación derivadas de las Historias de Usuarios, permitieron obtener un mayor nivel de detalle de cada una de las principales clases que intervienen en le desarrollo del componente. La organización de las misma en etapas o iteraciones permitió al equipo de desarrollo dar cumplimiento a los plazos y fechas definidas durante la planeación del proyecto. Las pruebas y validaciones realizadas permitieron evaluar la calidad y fiabilidad del componente desarrollado corroborando el cumplimiento de los requisitos funcionales y no funcionales identificados por lo que se cumple con los objetivos que se perseguían con su desarrollo.

Conclusiones Generales

Luego de cumplir con todas las tareas de investigación propuesta al inicio y con los objetivos trazados en el presente trabajo se concluyó que:

- A partir del estudio realizado sobre el uso de las recetas en distintos sistemas SCADA quedó evidenciada la necesidad de introducir en el SCADA GALBA la gestión de recetas.
- Con el análisis de las estructuras del módulo de aplicaciones del SCADA GALBA, se desarrolló un conjunto de funcionalidades que incluyen la gestión de parámetros de fórmula, gestión de plantillas de recetas y la gestión de recetas. Estas funcionalidades son de vital importancia en aras de dotar al planificador de tareas del GALBA de una herramienta para configuración y manejo de recetas básicas.
- Las pruebas y validaciones realizados a la herramienta permitieron evaluar su calidad y fiabilidad, demostrando su factibilidad de uso en el manejo de recetas.

Recomendaciones

Los objetivos pretendidos con el presente trabajo de diploma han sido logrados, pero a lo largo de su desarrollo, han ido surgiendo una serie de ideas que en un futuro podrían llevarse a cabo otorgándole a la misma una mayor utilidad y efectividad en lo que refiere al manejo de los procesos por lotes, para lo cual se recomienda:

- Integrar el presente componente al *plugin* de Aplicaciones del actual HMI del SCADA GALBA.
- Permitir exportar las recetas a *scripts* de python que puedan ser utilizadas por las tareas del módulo Aplicaciones.

Referencias Bibliográficas

[1].**Dictionary.com[en línea]**..http://dictionary.reference.com/2015

[2]Varios autores.. **Sistema de Visualización**., 2006.

[3]Xuletas. **OPC** http://www.xuletas.es/ficha/opc-2/. 2009

[4]Calviño Cruz, Yaniel, Kerton Martinez, Luanner , Socorro Borges Miguel. Ángel. **Ambiente de configuración para el sistema SCADA Guardián del Alba**.http://publicaciones.uci.cu/index.php/SC/article/viewFile/1285/705. 2013

[5]Crespo, Wiliam.¿**Que es la Automatización Industrial**?.

https://automatizacionindustrial.wordpress.com/2011/02/09/queeslaautomatizacionindustrial. 2011.

[6]Frenos, David. **Proceso productivo continuo y por lotes**. http://es.scribd.com/doc/58965722/Proceso-productivo-continuo-y-por-lotes . 2011

[7]Serna, Werned Yamid, Vergara, Diana Cecilia, Flórez, Juan Fernando. **Procedimiento de Modelado ISA 88 para ejecución de órdenes de producción basadas en Recipes.** 2011

[8]**Msc.** Herrera Vázquez, Moisés, **Msc.** Hernández Hernández, Yaneisy. **Especificación de Requisitos Suplementaria de Recetas.** 2013

[9]AMERICAN NATIONAL STANDARD. **Batch Control Part 1: Models and Terminolog.** 2011

[10]Copa-Data. **Zenon: Sistema HMI/SCADA y plataforma de comunicación[en línea]**..http://www.copadata.com/es/productos/product-features/zenon-solucion-y-sistema-hmiscada-copa-data.html. 2014

[11]Copa-Data. **Control de lotes compatible con ISA-88 en zenon[en línea]**..http://www.copadata.com/no_cache/es/productos/food-beverage/control-de-lotes-compatible-con-isa-88-en-zenon.html. 2013

[12]AG, Siemens. **SIMATIC WinCC Visualización de procesos con Plant Intelligence** 2008

[13]**Simatic WinCC/SES V7.3 para procesos basados en recetas y en secuencia**..http://www.infoplc.net/noticias/item/102373-simatic-wincc-ses-procesos-recetas-secuencia. 2015

[14].**DEFINICION DE METODOLOGIA**..https://es.scribd.com/doc/46558445/DEFINICION-DE-METODOLOGIA.2013

[15]Letelier, Patricio, Penades, María del Carmen. **Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP).** 2012

[16]Grady Booch, Ivar Jacobson, Rumbaugh, James. **El Lenguaje Unificado de Modelado. Manual de Referencia.** 2007

[17]Ecured. **Herramienta CASE**..http://www.ecured.cu/index.php/Herramienta_CASE. 2013.

[18]Stroustrup's, Bjarne. **The C++ Programming Language.** 1997

[19]**Qt Creator[en línea].** https://wiki.qt.io/Category:Tools::QtCreator_Spanish. 2015

[20]Fontanarossa,Diego, Garderes, Javier. **Gestión de Software. Extreme Programming (XP).** 2006

[21]Casas, Sandra, Reinaga, Hector. **Aspectos Tempranos: un enfoque basado en Tarjetas CRC**. 2009

[22]Cillero, Manuel. **Diagrama de Paquetes [en línea].** 2013

[23]**Ing**. Mengana Claro, Yuremis, **Ing**. Nuñez Alonso.,Arian Antonio. **Planificador de tareas para SCADA GALBA.** 2011

[24]Aedo Cuevas Ignacio. **Ingenieria de La Web y Patrones de Diseño[en línea]**..http://dialnet.unirioja.es/servlet/libro?codigo=323311. 2005

[25]Arias Calleja, Manuel. **Carmen. Estándares de codificación.**2007

[26]Chávez Lorenzo, Ariel **Estándares de codificación para C++.** 2011

[27]J.J. Gutiérrez, M, Escalona,J, Mejías, Torres J. **PRUEBAS DEL SISTEMA EN PROGRAMACIÓN EXTREMA.** 2014

Bibliografía

Herrera Vázquez, Moisés, **Hernández Hernández, Yaneisy**. Especificación de Requisitos Suplementaria de Recetas**.** 2013

AMERICAN NATIONAL STANDARD. Batch Control Part 1: Models and Terminolog. 2011

AMERICAN NATIONAL STANDARD. Batch Control Part 3: General and Site Recipe Models and Representation. 2003

Serna, Werned Yamid, **Vergara, Diana Cecilia**, **Flórez, Juan Fernando**. .Procedimiento de Modelado ISA 88 para ejecución de órdenes de producción basadas en Recipes. 2011

Mengana Claro, Yuremis, **Nuñez Alonso, Arian Antonio**. Planificador de tareas para SCADA GALBA. 2011

Letelier, Patricio, **Penades, María del Carmen**. Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP)**.** 2012

Fontanarossa, Diego, **Garderes, Javier**. Gestión de Software. Extreme Programming (XP). 2006

Calviño Cruz, Yaniel, **Kerton Martinez, Luanner** , **Socorro Borges Miguel Ángel.** Ambiente de configuración para el sistema SCADA Guardián del Alba. http://publicaciones.uci.cu/index.php/SC/article/viewFile/1285/705. 2013

Casas, Sandra, **Reinaga, Hector**. Aspectos Tempranos: un enfoque basado en Tarjetas CRC. 2009

Glosario de Términos

Módulos de Equipo: Según ISA-88, representa una colección funcional de equipos y/o módulos de control que responden a funciones determinadas. Una o varias piezas de equipo que pueden llevar a cabo un número finito de tareas específicas. Físicamente, pueden estar formados por módulos de control y otros módulos de equipo; deben ser parte de una unidad.

Módulo de Control. Según ISA-88, representa el nivel más bajo de la organización física de un proyecto. Puede estar formado por entidades que definan un lazo de control, un sensor de temperatura, un actuador, etc. Son los equipos que llevan a cabo las acciones de control básico. Típicamente, son una colección de sensores, válvulas, motores, actuadores y otros módulos de control que permiten establecer y mantener un estado específico de los equipos y procesos.

Framework: Estructura de software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación.

GUI: Interfaz Gráfica de Usuario.

HMI: *Human Machine Interface* (Interfaz hombre máquina).

IDE: *Integrated Development Environment* (Ambiente de desarrollo integrado).

Interfaz de Usuario: Medio que el usuario utiliza para comunicarse con una máquina, un equipo o una computadora, y comprende todos los puntos de contacto entre el usuario y el equipo, normalmente suelen ser fáciles de entender y fáciles de accionar.

Interfaz Gráfica de Usuario: Componente de una aplicación informática que el usuario visualiza y a través de la cual opera con ella. Está formada por ventanas, botones, menús e iconos, entre otros elementos.

Multiplataforma: Término utilizado frecuentemente en informática para indicar la capacidad o características de poder funcionar o mantener una interoperabilidad de forma similar en diferentes sistemas operativos o plataformas.

Punto: Los puntos se refieren a las direcciones de memoria que se configuran en el SCADA y que referencian a una variable capturada por un dispositivo de campo, o una variable del proceso. Pueden ser de tipo analógico, digital, texto y arreglo.

Logs: Registro oficial de eventos durante un periodo de tiempo en particular. Un log es usado para registrar datos o información sobre quién, qué, cuándo, dónde y por qué de un evento que ocurre para un dispositivo en particular o aplicación.

Anexos

Anexo1: Historias de Usuarios.

Historia de Usuario	
Número: 2	**Nombre de HU**:Gestionar Plantilla de Receta.
Fecha: 10-02-2015	**Usuario: Administrador**
Prioridad en el Negocio: Alta	**Riesgo de Desarrollo**: Medio
Tipo de actividad: nueva	**Iteración Asignada:1**
Puntos Estimados:1	**Puntos Reales:** 2
Descripción: El sistema debe ser capaz crear, modificar , mostrar y eliminar plantillas de recetas. En este paso se define todos los atributos de las recetas y los parámetros de la fórmula, enlazando cada parámetro con la variable del SCADA que le corresponde en el proceso.	
Observaciones: Las plantillas serán creadas desde la interfaz de configuración y persistirán en la base de datos de configuración	

Historia de Usuario	
Número: 6	**Nombre de HU:** Activar Receta
Fecha: 10-02-2015	**Usuario**: Operador de planta
Prioridad en el Negocio: Alta	Riesgo de Desarrollo: Alto
Tipo de actividad: nueva	Iteración Asignada:3
Puntos Estimados: 2	Puntos Reales: 3

Descripción: Esta acción implica que los valores de cada parámetro se escriban hacia las variables del sistema que están relacionadas. El operador selecciona la a receta utilizar y elije la acción aplicar, inmediatamente el sistema escribe los valores correspondiente de cada parámetro a las variable enlazadas. Una vez que se termina la escritura de cada parámetro a cada variable enlazada el operador recibe un mensaje que indica que la receta ha sido activada correctamente, en caso contrario se emite un error con el detalle del fallo. Al ejecutarse el comando correctamente se actualiza el atributo de estado de la receta a “Activa” además de modificar la fecha de su último uso.
Observaciones: Se debe crear un log para históricos que contenga el ID de la receta y el operador que la activó.

Historia de Usuario		
Número: 3	**Nombre de HU**: Gestionar Receta	
Fecha: 10-02-2015	**Usuario**: Operador de planta	
Prioridad en el Negocio: Alta		**Riesgo de Desarrollo: Medio**
Tipo de actividad: nueva		**Iteración Asignada:**2
Puntos Estimados: 2		**Puntos Reales:** 2
Descripción:El sistema debe ser capaz de crear, modificar, guardar y eliminar recetas a partir de las plantillas previamente configuradas.		
Observaciones:		

Anexo 2 Tarjetas CRC

ExecutorManagerRecipe
Súper Clase: Singleton

Responsabilidad	**Colaboración**
Clase única encargada de manejar las recetas.	ExecutorRecipe Parameter RecipeTemplate

ExecutorManagerTemplate	
Súper Clase: Singleton	
Responsabilidad	**Colaboración**
Clase única encargada de manejar las plantillas de recetas.	*Parameter* *RecipeTemplate*

ExecutorManagerParameter	
Súper Clase: Singleton	
Responsabilidad	**Colaboración**
Clase única encargada de manejar los parámetros de fórmula.	*Parameter*

ExecutorRecipe	
Súper Clase: Thread, Singleton	
Responsabilidad	**Colaboración**
Clase única encargada de ejecutar las recetas.	*Exchange* *Mutex* *Condition* *ExecuteQueue*

Anexo3: Tareas Ingeniería.

Tarea de Ingeniería	
No. de tarea: 4	**No. de HU:**1
Nombre de tarea: Eliminar parámetro de fórmula	
Tipo de tarea: Implementación	**Puntos estimados:** 0,2
Fecha inicio: 13:03:2015	**Fecha fin:** 16:03:2015
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Funcionalidad encargada de eliminar los parámetros de fórmula existentes, siempre y cuando los mismos no estén siendo usados en un receta.	

Tarea de Ingeniería	
No. de tarea: 5	**No. de HU:**1
Nombre de tarea: Validar los datos introducidos por el usuario	
Tipo de tarea: Implementación	**Puntos estimados:** 0,4
Fecha inicio: 17:03:2015	**Fecha fin:** 19:03:2015
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Funcionalidad encargada de comprobar y validar que los datos introducidos por el usuario sean correctos y cumplan con las condiciones definidas por el cliente.	

Tarea de Ingeniería	
No. de tarea: 6	**No. de HU:** 2
Nombre de tarea: Diseñar interfaz de una extensión para el planificador de tareas que permita la gestión de plantillas.	
Tipo de tarea: Diseño	**Puntos estimados:** 0,2
Fecha inicio: 27:03:2015	**Fecha fin:** 27:03:2015
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Tiene como fin, crear la interfaz de la herramienta que se integrará al planificador de tareas para la gestión de la lógica operacional.	

Tarea de Ingeniería	
No. de tarea: 7	**No. de HU:** 2
Nombre de tarea: Crear plantilla de recetas	
Tipo de tarea: Implementación	**Puntos estimados:** 0,4

Fecha inicio:	**Fecha fin**
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Funcionalidad encargada de crear las plantilla de receta, definiendo todos los atributos de las plantillas y los parámetros de la fórmula.	

Tarea de Ingeniería	
No. de tarea: 8	**No. de HU:** 2
Nombre de tarea: Modificar plantilla de recetas	
Tipo de tarea: Implementación	**Puntos estimados:** 0,4
Fecha inicio: 28:03:2015	**Fecha fin:** 30:03:2015
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Funcionalidad encargada de modificar las plantillas de recetas. Si las plantillas tienes recetas asociadas no debe permitir modificar la lista de parámetros.	

Tarea de Ingeniería	
No. de tarea: 9	**No. de HU:** 2
Nombre de tarea: Eliminar plantilla de recetas	
Tipo de tarea: Implementación	**Puntos estimados:** 0,2
Fecha inicio: 01:04:2015	**Fecha fin:** 02:04:2015
Programador responsable: Yosvany Leyva Pizarrosa	
Descripción: Funcionalidad encargada de eliminar las plantilla de recetas existentes, siempre y cuando las mismas no contengan recetas que hayan sido creadas a partir de ellas.	

Anexo 4: Casos de pruebas.

<table>
<tr><th colspan="3">Caso de Prueba de Aceptación</th></tr>
<tr><td>Número: 2</td><td colspan="2">Historia de usuario: Modificar parámetro de fórmula</td></tr>
<tr><td colspan="3">Probador encargado: Yosvany Leyva Pizarrosa</td></tr>
<tr><td colspan="3">Descripción: Permitir que el usuario modifique un parámetro de fórmula, a través de la interfaz visual.</td></tr>
<tr><td colspan="3">Condiciones de ejecución:
1. El parámetro a modificar debe existir en el sistema.
2. El usuario que interactúa con el sistema debe tener privilegios para realizar la operación modificar.</td></tr>
<tr><td colspan="3">Entrada/Pasos de ejecución:
1. El usuario selecciona el parámetro a modificar accionando doble clic sobre el mismo o seleccionado el botón modificar.
2. El usuario selecciona botón “Aceptar”.</td></tr>
<tr><th>Casos válidos</th><th>Resultado esperado</th><th>Resultado de prueba</th></tr>
<tr><td>Insertar los datos que desea modificar (nombre, descripción o valor) o seleccionar los campos que desea modificar (unidad de medida o tipo de datos). El usuario no pude dejar campos vacíos.</td><td>Luego de modificar los campos estimados por el usuario, si el proceso ha resultado correcto se actualiza la base de datos con los nuevos datos del parámetro</td><td>Satisfactorio</td></tr>
<tr><th>Casos no válidos</th><th>Resultado esperado</th><th>Resultado de prueba</th></tr>
<tr><td>Caracteres alfabéticos en campos que no lo requieran (valor).</td><td>Se debe mostrar un mensaje de error que se han introducido</td><td>Satisfactorio</td></tr>
</table>

	caracteres inválidos.	
Evaluación de la prueba: Satisfactorio		

Caso de Prueba de Aceptación		
Número: 3	**Historia de usuario:** Eliminar parámetro de fórmula.	
Probador encargado: Yosvany Leyva Pizarrosa		
Descripción: Permitir que el usuario elimine el parámetro de fórmula que desee a través de la interfaz visual.		
Condiciones de ejecución: 1. El usuario que interactúa con el sistema debe tener privilegios para realizar la operación eliminar (administrador). 2. El parámetro no puede estar siendo utilizado en la fórmula simplificada de una receta.		
Entrada/Pasos de ejecución: 1. El usuario selecciona el parámetro a eliminar accionando un clic sobre el mismo. 2. El usuario selecciona el botón “Eliminar parámetro” que se encuentra en el buscador de parámetros.		
Casos Válidos	**Resultado esperado**	**Resultado de prueba**
Seleccionar el identificador del parámetro que se desea eliminar y posteriormente se selecciona el botón “eliminar parámetro”.	Luego de eliminar el parámetro se actualizan la base de datos y la vista con los parámetros restantes	Satisfactorio
Casos no válidos	**Resultado esperado**	**Resultado de prueba**
No seleccionar el identificador del parámetro a eliminar. El parámetro está	Se le mostrará un mensaje de error al usuario indicando el tipo	Satisfactorio

asociado a una receta o varias recetas.	de error ocurrido.	
Evaluación de la prueba: Satisfactorio		

Caso de Prueba de Aceptación		
Número: 4	**Historia de usuario:** Crear plantilla de recetas	
Probador encargado: Yosvany Leyva Pizarrosa		
Descripción: Se comprueba la funcionalidad crear plantilla a través de la interfaz visual.		
Condiciones de ejecución: 1. El usuario que interactúa con el sistema debe tener privilegios para realizar la operación crear plantilla (administrador).		
Entrada/Pasos de ejecución: 1. El usuario selecciona la opción crear nueva plantilla, que aparece en el menú principal de la y el acceso directo de la barra de herramientas. 2. El usuario selecciona los parámetros que tendrá la fórmula simplificada de la receta. 3. El usuario selecciona botón “Aceptar”.		
Casos válidos	**Resultado esperado**	**Resultado de prueba**
Insertar nombre de la plantilla , insertar descripción de la misma, adicionar los parámetros de fórmula. Debe tenerse en cuenta que todos los campos son obligatorios.	Luego de adicionar la plantilla, si el proceso ha resultado correcto se actualiza la base de datos con la nueva plantilla.	Satisfactorio
Casos no válidos	**Resultado esperado**	**Resultado de prueba**
La plantilla se encuentra insertada en el	Se debe mostrar un error	Satisfactorio

sistema. Caracteres extraños en campos que no lo requieran (nombre).	indicando que existe una plantilla con igual nombre.	
Evaluación de la prueba: Satisfactorio		

Caso de Prueba de Aceptación		
Número: 5	**Historia de usuario:** Modificar plantilla de recetas	
Probador encargado: Yosvany Leyva Pizarrosa		
Descripción: Permitir que el usuario modifique una plantilla de recetas a través de la interfaz visual.		
Condiciones de ejecución: 1. El usuario que interactúa con el sistema debe tener privilegios para ejecutar la operación modificar plantilla de recetas (administrador).		
Entrada/Pasos de ejecución: 1. El usuario selecciona la plantilla de receta a modificar accionando doble clic sobre la misma. 2. El usuario selecciona botón “Aceptar”.		
Casos Válidos	**Resultado esperado**	**Resultado de prueba**
Insertar los datos que desea modificar (nombre, descripción). El usuario no pude dejar campos vacíos.	Luego de modificar los campos estimados por el usuario, si el proceso ha resultado correcto se actualizan la base de datos y la vista con los nuevos datos de la plantilla de receta.	Satisfactorio
Casos no válidos	**Resultado esperado**	**Resultado de prueba**
Sustituir el nombre de la plantilla de receta por otro nombre igual al de una	Se debe mostrar un error indicando que existe una plantilla	Satisfactorio

plantilla ya existente en el sistema.	con igual nombre	
Evaluación de la prueba: Satisfactorio		

Caso de Prueba de Aceptación		
Número: 6	**Historia de usuario:** Eliminar plantilla de recetas	
Probador encargado: Yosvany Leyva Pizarrosa		
Descripción: Permitir que el usuario elimine la plantilla de recetas que desee a través de la interfaz visual.		
Condiciones de ejecución: 1. El usuario que interactúa con el sistema debe tener los privilegios para ejecutar la acción eliminar plantilla de receta (administrador).		
Entrada/Pasos de ejecución: 1. El usuario selecciona la plantilla de receta a eliminar accionando un clic sobre la misma. 2. El usuario selecciona el botón “Eliminar” que se encuentra en la barra de herramientas.		
Casos Válidos	**Resultado esperado**	**Resultado de prueba**
Seleccionar el identificador de la plantilla de recetas que se desea eliminar y posteriormente se seleccionar el botón “eliminar”.	Luego de eliminar la plantilla de recetas se actualizan la base de datos y la vista con las plantillas de recetas restantes.	Satisfactorio
Casos no válidos	**Resultado esperado**	**Resultado de prueba**
No seleccionar el identificador de la plantilla de recetas a eliminar. La plantilla de recetas está asociado a una o varias recetas.	Se le mostrará un mensaje de error al usuario indicando el tipo de error ocurrido.	Satisfactorio

Evaluación de la prueba: Satisfactorio

Caso de Prueba de Aceptación		
Número: 7	**Historia de usuario:** Crear receta.	
Probador encargado: Yosvany Leyva Pizarrosa		
Descripción: Se comprueba la funcionalidad crear plantilla a través de la interfaz visual.		
Condiciones de ejecución: 1. Deben existir plantillas de recetas previamente creadas en el sistema. 2. El usuario que interactúa con el sistema debe tener privilegios para realizar la operación crear receta (administrador u operador).		
Entrada/Pasos de ejecución: 1. El usuario selecciona la opción crear nueva receta, que aparece en el menú principal de la y el acceso directo de la barra de herramientas. 2. El usuario selecciona los parámetros (valor) que tendrá la fórmula simplificada de la receta. 3. El usuario selecciona el botón “Aceptar”.		
Casos Válidos	**Resultado esperado**	**Resultado de prueba**
Insertar nombre de la receta, insertar descripción de la misma, seleccionar el estado de la receta y modificar los parámetros de fórmula. Debe tenerse en cuenta que todos los campos son obligatorios.	Luego de adicionar la receta, si el proceso ha resultado correcto se actualizan la base de datos y la vista con la nueva receta.	Satisfactorio
Casos no válidos	**Resultado esperado**	**Resultado de prueba**

La receta se encuentra insertada en el sistema. Caracteres extraños en campos que no lo requieran (nombre). Dejar algún campo vacío.	Se debe mostrar un error indicando que existe una receta con igual nombre o que el campo X no puede estar vacío.	Satisfactorio
Evaluación de la prueba: Satisfactorio		

Caso de Prueba de Aceptación	
Número: 8	**Historia de usuario:** Eliminar receta
Probador encargado: Yosvany Leyva Pizarrosa	
Descripción: Permitir que el usuario elimine la receta que desee a través de la interfaz visual.	
Condiciones de ejecución: 1. El usuario que interactúa con el sistema debe tener los privilegios para ejecutar la acción eliminar receta (administrador).	
Entrada/Pasos de ejecución: 1. El usuario selecciona la receta a eliminar accionando un clic sobre la misma. 2. El usuario selecciona el botón “Eliminar” que se encuentra en la barra de herramientas.	

Casos Válidos	**Resultado esperado**	**Resultado de prueba**
Seleccionar el identificador de la receta que se desea eliminar y posteriormente se seleccionar el botón “eliminar”.	Luego de eliminar la receta se actualizan la base de datos y la vista con las recetas restantes.	Satisfactorio
Casos no válidos	**Resultado esperado**	**Resultado de prueba**

No seleccionar el identificador de la receta a eliminar. La receta a eliminar se encuentra en ejecución o en espera para ser ejecutada.	Se le mostrará un mensaje de error al usuario indicando el tipo de error ocurrido.	Satisfactorio
Evaluación de la prueba: Satisfactorio.		