

ESCUELA POLITÉCNICA DEL EJÉRCITO

DPTO. DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

MÉTODO ÁGIL SCRUM, APLICADO A LA  
IMPLANTACIÓN DE UN SISTEMA INFORMÁTICO PARA  
EL PROCESO DE RECOLECCIÓN MASIVA DE  
INFORMACIÓN CON TECNOLOGÍA MÓVIL

Previa la obtención del Título de:

INGENIERO EN SISTEMAS E INFORMÁTICA

POR:

KLEBER MANUEL TOAPANTA CHANCUSI

Sangolquí, noviembre 2012

## **CERTIFICACIÓN**

Certifico que el presente trabajo fue realizado en su totalidad por el Sr. Kléber Manuel Toapanta Chancusi como requerimiento parcial a la obtención del título de INGENIERO EN SISTEMAS E INFORMÁTICA.

Noviembre del 2012

---

ING. MARCO VERGARA

DIRECTOR

## LISTADO DE TABLAS

Tabla 1. Matriz de Soluciones, diagrama Causa–Efecto (Parte I) .....	6
Tabla 2. Matriz de Soluciones, diagrama Causa–Efecto (Parte II) .....	7
Tabla 3. Comparación entre metodologías ágiles y tradicionales .....	27
Tabla 4. Roles del SCRUM .....	32
Tabla 5. Técnicas de pruebas .....	53
Tabla 6. Simbología utilizada en los diagramas .....	65
Tabla 7. Requerimientos Funcionales/No Funcionales – Sprint 0 .....	66
Tabla 8. Equipo de Trabajo y Roles .....	70
Tabla 9. Backlog Producto .....	71
Tabla 10. Requerimientos Funcionales/No Funcionales- Sprint1 .....	75
Tabla 11. Historias de Usuario - Sprint 1 .....	76
Tabla 12. Actores del sistema .....	77
Tabla 13. Especificación del caso de uso: Gestionar Usuarios .....	79
Tabla 14. Especificación del caso de uso: Gestionar Equipos .....	80
Tabla 15. Especificación del caso de uso: Seleccionar Usuario .....	81
Tabla 16. Especificación del caso de uso: Asignar/Reasignar Responsable ....	82
Tabla 17. Especificación del caso de uso: Seleccionar Perfil .....	83
Tabla 18. Especificación del caso de uso: Seleccionar Formulario .....	83
Tabla 19. Especificación del caso de uso: Gestionar Perfiles .....	84
Tabla 20. Especificación del caso de uso: Gestionar Formularios .....	85
Tabla 21. Especificación del caso de uso: Definir Usuarios del perfil .....	86
Tabla 22. Especificación del caso de uso: Definir Formularios del perfil .....	87
Tabla 23. Especificación del caso de uso: Gestión de Catálogos .....	88
Tabla 24. Backlog Sprint “Módulo de Administración” .....	92
Tabla 25. Requerimientos Funcionales/No Funcionales- Sprint 2 .....	95
Tabla 26. Historias de Usuarios – Sprint 2 .....	96
Tabla 27. Actores del sistema .....	97
Tabla 28. Especificación del caso de uso: Carga Planes de trabajo .....	99
Tabla 29. Formato de archivo para intercambio magnético .....	100
Tabla 30. Especificación del caso de uso: Consultar Plan de trabajo .....	101
Tabla 31. Especificación del caso de uso: Seleccionar Lecturas .....	102
Tabla 32. Especificación del caso de uso: Asignar/Reasignar Lecturas .....	102

Tabla 33.Especificación del caso de uso: Consultar Asignación.....	103
Tabla 34.Especificación del caso de uso: Consultar estado Sincronización ..	104
Tabla 35.Requerimientos Funcionales/No Funcionales- Sprint 3.....	110
Tabla 36.Historias de Usuarios – Sprint 3 .....	111
Tabla 37.Especificación del caso de uso: Login App Móvil .....	113
Tabla 38.Especificación del caso de uso: Mantenimiento Aplicación móvil....	114
Tabla 39.Especificación del caso de uso: Seleccionar Lectura.....	115
Tabla 40.Especificación del caso de uso: Tomar Lectura .....	115
Tabla 41.Códigos de novedades.....	117
Tabla 42.Ejemplo Validación Giro .....	118
Tabla 43.Especificación del caso de uso: Desbloquear Lectura .....	118
Tabla 44.Especificación del caso de uso: Registrar Excedente .....	119
Tabla 45.Requerimientos Funcionales/No Funcionales- Sprint 4.....	126
Tabla 46. Historia de usuario “Sincronización” - Alcance .....	126
Tabla 47. COPIA - Especificación del caso de uso: Sincronizar Lecturas en Línea .....	128

## LISTADO DE FIGURAS

Figura 1. Diagrama Causa-Efecto. Análisis proceso recolección masiva de información para ASISTECOM.....	5
Figura 2. Rational Unified Process .....	16
Figura 3. MICROSOFT SOLUTION FRAMEWORK.....	17
Figura 4. SCRUM .....	21
Figura 5. Extreme Programming .....	22
Figura 6. Crystal Methodologies.....	23
Figura 7. Dynamic Systems Development Method.....	24
Figura 8. Feature -DrivenDevelopment .....	25
Figura 9. Lean Development .....	26
Figura 10. Elementos del SCRUM .....	31
Figura 11. SPRINT .....	35
Figura 12. Builds continuos .....	39
Figura 13.Modelo V .....	51
Figura 14.El modelo W .....	51
Figura 15.Modelo de Pruebas Orientada a Objetos para el Ciclo de Vida Completo.....	52
Figura 16.Plantilla para especificación de casos de uso.....	57
Figura 17.Ejemplo Diagrama Entidad Relación.....	58
Figura 18. Definiciones Sprintometer .....	59
Figura 19.Herramienta de desarrollo VS2010.....	60
Figura 20.Proceso de Lectura de medidores de consumo IN SITU .....	62
Figura 21 - Diagrama de procesos .....	64
Figura 22.Escala Importancia Definida por Product Owner.....	71
Figura 23.Modelo de Datos del Sistema.....	72
Figura 24. Arquitectura Aplicación.....	74
Figura 25.Casos de uso - Administración.....	78
Figura 26.Diagrama Entidad Relación-Módulo Administración .....	89
Figura 27.Arquitectura Aplicación de escritorio – Módulo de Administración ...	91
Figura 28. Componentes - Módulo Administración.....	91
Figura 29. Backlog Sprint 1- Sprint to Meter.....	93

Figura 30.Reporte de errores – JIRA .....	94
Figura 31.Casos de Uso - Módulo Gestión.....	98
Figura 32.Diagrama Entidad Relación-Módulo Gestión.....	105
Figura 33. Arquitectura Aplicación - Sección Aplicación de escritorio .....	107
Figura 34.Componentes - Módulo Gestión.....	107
Figura 35.Diagramas de casos de uso - Aplicación Móvil .....	112
Figura 36. Modelo de datos - Aplicación Móvil. ....	120
Figura 37.Arquitectura Aplicación Móvil .....	121
Figura 38. Historia de Usuario “Sincronización en línea”.....	127
Figura 39. COPIA - Diagrama Entidad Relación-Aplicación Móvil.....	129
Figura 40. Componentes a exponer en Servicio WCF .....	130
Figura 41. Diagrama Físico SISTEMA.....	131
Figura 42. Método SCRUM adaptado al proyecto R.M.I con dispositivos móviles .....	134
Figura 43.Ciclo de vida en cascada en cada Sprint .....	135
Figura 44.Ejecución proyecto R.M.I con dispositivos móviles .....	135
Figura 45. Ejemplo buenas prácticas programación .....	138

## LISTADO DE ANEXOS

ANEXO A - PRUEBAS REALIZADAS EN LA ETAPA DE CODIFICACIÓN ...	144
ANEXO B - GLOSARIO DE TÉRMINOS Y ABREVIATURAS.....	153
ANEXO C - ACTAS DE REUNIONES CON PRODUCT OWNER.....	155
ANEXO D - MANUAL DE USUARIO .....	174
ANEXO E - CARTA DE AUSPICIO Y ACEPTACIÓN.....	193

## ÍNDICE DE CONTENIDO

CAPÍTULO 1 .....	3
1.1. Introducción .....	3
1.2. Planteamiento del problema .....	5
1.3. Alcance .....	9
1.4. Objetivos .....	11
CAPÍTULO 2 .....	13
2.1. Marco Teórico .....	13
2.2. Metodologías Tradicionales de Desarrollo de Software .....	14
2.2.1. Principales metodologías Tradicionales .....	15
2.3. Metodologías Ágiles de Desarrollo de Software.....	17
2.3.2. El manifiesto Ágil .....	18
2.3.3. Principales metodologías Ágiles .....	21
2.4. Metodologías Ágiles vs. Tradicionales .....	26
2.5. SCRUM .....	28
2.5.1. Introducción .....	28
2.5.2. La Esencia De SCRUM .....	29
2.5.3. Elementos del SCRUM.....	30
CAPÍTULO 3 .....	42
3.1. Metodología.....	42
3.2. Desarrollo iterativo e incremental .....	43
3.3. Etapas Del Proceso De Desarrollo.....	44
3.3.1. Planificación.....	45
3.3.2. Análisis .....	46
3.3.3. Diseño .....	46
3.3.4. Construcción y Pruebas.....	47



3.3.5. Implementación .....	55
3.4. Herramientas .....	56
3.4.1. Técnicas de relevamiento .....	56
3.4.2. Casos de Uso .....	56
3.4.3. Diagrama de Entidad Relación (DER) .....	57
3.4.4. Sprintometer .....	58
3.4.5. Visual Studio 2010.....	59
3.4.6. Jira.....	60
CAPÍTULO 4 .....	61
4.1. Ejecución del Proyecto.....	61
4.1.1. Planificación.....	61
4.1.2. Alcance del software.....	68
4.1.3. Conformación del equipo de trabajo .....	69
4.1.4. Definición del Backlog del Producto. ....	70
4.1.5. Diseño .....	72
4.1.6. Sprint1 “Módulo de Administración” .....	74
4.1.7. Sprint2 “Módulo de Gestión” .....	95
4.1.8. Sprint3 “Aplicación Móvil” .....	110
4.1.9. Sprint4 “Sincronización en Línea” .....	125
4.2. Conclusiones.....	131
4.3. Recomendaciones.....	139
BIBLIOGRAFÍA .....	141
ANEXO A - PRUEBAS REALIZADAS EN LA ETAPA DE CODIFICACIÓN ...	144
Pruebas Sprint 1”Módulo de administración” .....	144
Pruebas Sprint 2”Módulo de Gestión” .....	149
Pruebas Sprint 3” Aplicación Móvil” .....	151
ANEXO B - GLOSARIO DE TÉRMINOS Y ABREVIATURAS.....	153

ANEXO C - ACTAS DE REUNIONES CON PRODUCT OWNER.....	155
Acta N°2 – Sprint 0 .....	155
Acta N°3 – Sprint 1 .....	158
Acta N°5 – Sprint 2 .....	163
Acta N°6 – Sprint 3 .....	169
ANEXO D - MANUAL DE USUARIO .....	174
Prerrequisitos .....	175
Instalación .....	175
Aplicación principal .....	177
Generalidades .....	177
Módulo de administración .....	181
Administración de usuarios.....	181
Administración de equipos.....	182
Asignación de equipos.....	182
Módulo de Gestión .....	184
Cargar planes de trabajo .....	184
Cambiar estado de los planes de trabajo.....	185
Asignación de lecturas.....	186
Sincronización de datos asignados .....	187
Rendir planes de trabajo.....	188
Aplicación móvil.....	189
Prerrequisitos .....	189
Instalación .....	189
Acceso a la aplicación.....	190
Gestión de lecturas en campo.....	190
DESBLOQUEO DE LECTURAS .....	191
Lecturas Excedentes.....	192

Sincronización en línea.....	192
ANEXO E - CARTA DE AUSPICIO Y ACEPTACIÓN.....	193
BIOGRAFÍA.....	194

## RESUMEN

El mercado actual es altamente competitivo y cambiante. En ese contexto el desarrollo del Software busca básicamente rapidez, calidad y reducción de costos en la ejecución de sus proyectos; para asumir estos retos es necesario tener agilidad y flexibilidad. Estas características se constituyen en el fundamento mismo de las metodologías ágiles de desarrollo.

En el ámbito de las metodologías de desarrollo de software existe un gran número de alternativas y los responsables de cada proyecto tienen la difícil tarea de seleccionar la alternativa que mejor se ajuste a sus necesidades y recursos.

El presente estudio se enfocó en el análisis del método ágil SCRUM para la implementación de una metodología aplicada al desarrollo de software para la recolección masiva de información con dispositivos móviles.

La ejecución y culminación del proyecto permitió establecer una metodología basada en SCRUM, complementada con otros métodos. El resultado es un producto de software funcional, en cuyo desarrollo se pudo demostrar la validez de SCRUM aplicado a proyectos de software de mediano tamaño, en entornos cambiantes, con grupos de trabajo pequeños que involucran permanentemente al dueño del producto.

## **ABSTRACT**

The current market is very competitive and always is changing. For this reason the development of the Software is looking for speed, quality and cost reduction in the execution of their projects, to take on these challenges is necessary to have agility and flexibility, these characteristics constitute the basis for agile development methodologies.

In the area of the methodologies used to address software development projects there are a lot of alternatives, and the responsible of each project must to select the best alternative according their needs and resources.

The project is focused on the study of agile method SCRUM and implementation of a methodology applied to software development for the massive collection of information with mobile devices, using SCRUM.

The execution and completion of the project allowed to establish a methodology based in SCRUM, supplemented with other methods. The result is a functional software product, that helped to demonstrate the effectiveness of SCRUM applied to software projects of medium size, in changing environments, with small groups of work constantly involving the product owner.

# CAPÍTULO 1

## 1.1. Introducción

Existen en el medio muchas empresas para las cuales la recolección de datos in situ es parte fundamental de la ejecución de sus procesos operativos, por ejemplo Eléctrica de Quito, Empresa Metropolitana de Agua Potable, Ilustre Municipio Quito, entre otras a nivel nacional; para las que el levantamiento de información (consumo mensual de medidores, información catastral, etc.) es indispensable en la ejecución de otros procesos como recaudación, facturación, análisis, registro, etc. y para los cuales es necesario una eficiente y oportuna recolección de datos in situ.

En ocasiones este proceso se podía llevar a cabo de una forma completamente manual, utilizando únicamente formularios y personal dedicado a llenarlos con la información requerida. En estos casos, el posterior procesamiento (manual o automático) de la información recolectada, puede presentar una alta probabilidad de obtener resultados incorrectos derivados de errores de caligrafía o de interpretación de los datos recolectados, errores de transcripción, pérdidas de información debido a los medios utilizados, entre otros.

Como respuesta a esta problemática y de la mano de los avances tecnológicos, surgen en el mercado, una gran variedad de dispositivos para la recolección de información in situ, los cuales disminuyen significativamente el riesgo de errores que se presentaban con un proceso manual. Estos dispositivos ofrecen un mayor volumen de trabajo y agilitan el procesamiento de la información recolectada,

puesto que esta puede transferirse desde y hacia algún sistema informático que la procese los datos.

Los primeros equipos recolectores de datos eran grandes, pesados, equipados con baterías de corta duración, arquitectura cerrada que limitaba su utilización, con funcionalidades bastante limitadas, todo esto sumado a los altos costos de adquisición y mantenimiento.

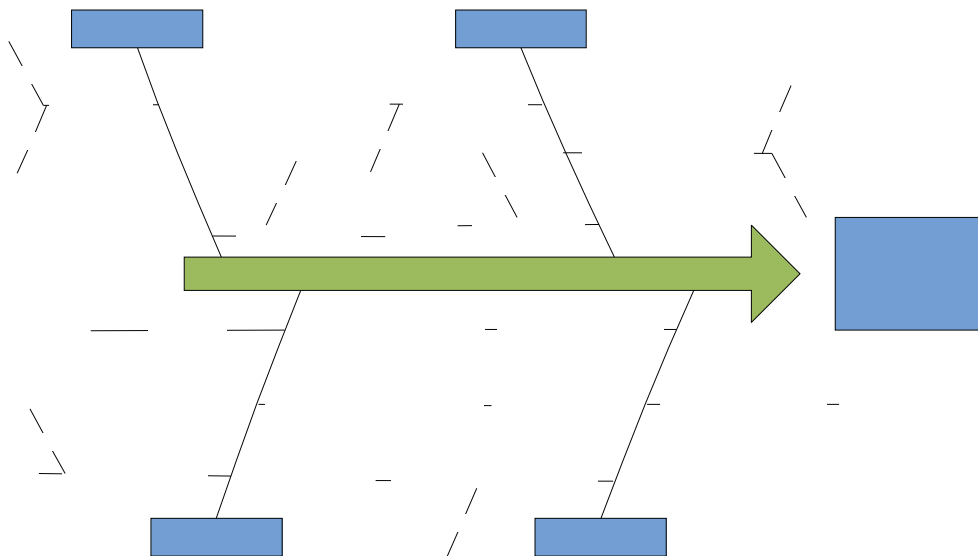
En la actualidad, los avances tecnológicos en equipos de telefonía móvil, su gran difusión, relativos bajos costos, entre otras características, hacen de estos una gran alternativa en el mejoramiento y optimización del proceso de recolección masiva de información in situ, especialmente por la posibilidad de transferir información en línea, agilitando enormemente el proceso.

En este contexto, ASISTECOM Cía. Ltda. , es una empresa dedicada a brindar servicios de asistencia técnica, financiera y comercial, enfocada especialmente a empresas de servicios básicos a nivel nacional. Brinda para algunos de sus clientes el servicio de recolección masiva de información in situ. Para ello, la empresa utiliza actualmente un sistema informático de propiedad de un proveedor externo. La creciente demanda de los servicios prestados por la empresa y la necesidad de ajustarse a los requerimientos particulares de cada uno de sus clientes, hace que la empresa ASISTECOM Cía. Ltda. decida auspiciar el presente proyecto , con la finalidad de obtener un herramienta informática propia, que le permita: gestionar en forma ágil, oportuna y eficiente el proceso de recolección masiva de información para uno de sus principales clientes (Empresa Eléctrica de Quito), aprovechar al máximo la tecnología disponible en el mercado tanto en el sector de la informática como de la tecnología móvil y hacer que la

nueva herramienta se convierta en poco tiempo en una alternativa viable al sistema informático actual.

## 1.2. Planteamiento del problema

Para exponer los factores que justifican la ejecución del presente proyecto, se usará un Diagrama Causa-Efecto<sup>1</sup> (o Espina de pescado). En este diagrama, se precisa un problema central o efecto, en este caso definido por el proceso de RECOLECCIÓN MASIVA DE INFORMACIÓN IN SITU; para este efecto o consecuencia, se identifican los factores causales más importantes representados por las espinas, como lo muestra la siguiente figura.



**Figura 1. Diagrama Causa-Efecto. Análisis proceso recolección masiva de información para ASISTECOM**

<sup>1</sup> Apuntes de Administración. (s.f.). *apuntesyama*. Recuperado el 10 de 02 de 2012, de <http://apuntesyama.galeon.com/ischikawa.html>



### 1.2.1. Matriz de soluciones

Una vez realizado un análisis de causa efecto, es necesario realizar una ponderación de la misma para identificar las soluciones que causan mayor impacto y tomar decisiones.

**Tabla 1. Matriz de Soluciones, diagrama Causa-Efecto (Parte I)**

CRITERIOS	PESO <sup>A</sup>	CAUSA	SUB CAUSAS	RESUMEN	ACCIONES / SOLUCIONES	PONDERACIÓN <sup>B</sup> 1-10	CUANTIFICACIÓN CRITERIOS	TOTAL <sup>C</sup>
Equipos	1	Subutilizados	(1) Nuevas tecnologías	Existen nuevas tecnologías, pero la empresa no puede hacer uso de estas	Utilización de nuevas tecnologías	7	16	16
			(0) Nuevos Equipos	Existen equipos más eficientes, de menor costo, pero la empresa no puede utilizarlos	Adquisición de Nuevos equipos	4		
		Soporte Técnico	(2) Frecuencia	Se recurre con alta frecuencia al soporte técnico del actual proveedor	Disminuir la necesidad de soporte técnico	5		
Software	2	Soporte Técnico	(5) Frecuencia	Se recurre con alta frecuencia al soporte técnico del actual proveedor	Disminuir la necesidad de soporte técnico	8	22	44
			Mantenimiento	(6) Cambios frecuentes	El proceso requiere con frecuencia de cambios en el Sistema Informático	Software propio, facilidades para mantenimiento en el software		
		Plataforma informática		(7) Tecnología en desuso	El actual proveedor obliga a la utilización de tecnología e desuso	Eliminar esta necesidad		
			(8) Sin soporte técnico	La tecnología en desuso, no tiene soporte técnico.	Utilizar tecnología de fácil acceso en el mercado.	2		

**Tabla 2. Matriz de Soluciones, diagrama Causa-Efecto (Parte II)**

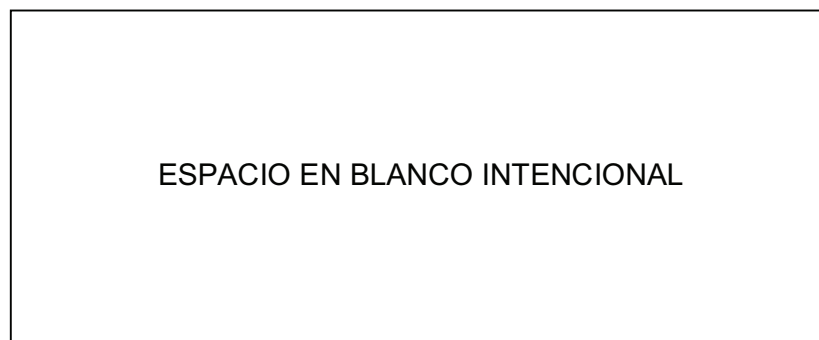
Personas	3	Tiempos de trabajo	(3) Procesos lentos	Existen procesos que son demasiado lentos con el proveedor actual.	Buscar alternativas a los procesos que toman demasiado tiempo.	9	14	42
		(12) Control		Se requiere mantener mayor control sobre el trabajo del personal.	Controlar de forma eficiente el trabajo del personal operativo	3		
		(4) Facilidad de Uso (software)		El software de recolección de datos debe ser de fácil utilización para el personal operativo.	Mejorar facilidad de uso	2		
Administración	4	Costos por soporte técnico	11 Excesivo	Gastos elevados por mantenimiento y soporte técnico.	Reducción de gastos por soporte técnico.	8	26	104
		(10) Calidad de servicio		Se busca ampliar la calidad del servicio que la empresa presta a sus clientes	Mejorar calidad de servicio que se ofrece a los clientes.	9		
		(9) Imagen institucional		La calidad del servicio influye en la imagen de la empresa.	Mejorar calidad para mejorar imagen institucional	9		

**Nota:** El valor más alto en la columna de ponderación de criterios es el indicador de mayor importancia para atacar el problema. Fuente: Elaboración propia.

<sup>a</sup> Importancia del criterio respecto del tema analizado.

<sup>b</sup> Ponderación de 0 a 10 en cada sub criterio, siendo 0 el calificativo de menor importancia y 10 el de mayor importancia.

<sup>c</sup> Sumatoria de las ponderaciones de cada sub criterio, agrupados en criterios y multiplicado por el peso del criterio.



### **1.2.2. Justificación**

Una vez analizado el proceso de recolección masiva de información in situ para la empresa ASISTECOM Cía. Ltda. , se determina que el principal objetivo es brindar un mejor servicio así como optimizar el proceso para reducir los gastos generados en su ejecución. Se determina entonces que las principales deficiencias en la ejecución del proceso se encuentran ligadas a problemas de tecnologías de información, por lo que es necesario enfocarse en resolver estos problemas para optimizar a la brevedad posible el mencionado proceso.

Para cumplir con las necesidades mencionadas, se plantea la utilización del Método Ágil de Desarrollo “SCRUM”, aplicado a la Implantación de un Sistema Informático para la Administración del Proceso Operativo en Recolección Masiva De Información Con Tecnología Móvil.

La ejecución de este proyecto permitirá a ASISTECOM Cía. Ltda., desarrollar e implantar en corto tiempo un sistema informático propio, establecer un marco de trabajo eficiente para el desarrollo y posterior mantenimiento, optimizar la utilización de recursos (Hardware, Software, Tecnología Móvil), reducir los gastos generados por mantenimiento de software y brindar un mejor y más eficiente servicio a sus clientes actuales y nuevos.

Los principales motivos que han llevado a la elección de SCRUM son:

- Es uno de los métodos de gestión de proyectos, de la denominados filosofía ágil, que permite un manejo apropiado de las expectativas del cliente, basadas en resultados tangibles.

- Es uno de los métodos ágiles menos conocidos y utilizados en nuestro medio, por lo que el presente trabajo puede resultar en un aporte significativo al conocimiento del mismo.
- Otro factor importante es la experiencia con que cuenta el responsable del presente trabajo en la ejecución de proyectos de desarrollo de software utilizando una variación la metodología seleccionada.

### **1.3. Alcance**

En este contexto, se establecen los aspectos que debe abarcar la ejecución del presente proyecto, así como los alcance del nuevo sistema informático para la Administración del Proceso de Recolección Masiva de Información In Situ con Tecnología Móvil para la empresa ASISTECOM Cía. Ltda., esto es:

- Estudio y aplicación del método Ágil “SCRUM” aplicado a la IMPLANTACIÓN de un nuevo sistema informático para ASISTECOM Cía. Ltda., el mismo que le permita gestionar el proceso de recolección masiva de información in situ para uno de sus principales clientes (EEQ – Lectura de consumo eléctrico).
- Establecer SCRUM como el marco de trabajo oficial de ASISTECOM para el desarrollo de nuevas funcionalidades y/o mantenimiento de la aplicación, que no se ha establecido como alcance de este proyecto.
- Análisis, desarrollo e implementación del sistema informático comprendido por:

- Una Aplicación de escritorio con los siguientes módulos:
  - Módulo de Administración de Seguridad: Crear, eliminar, modificar usuarios, equipos y perfiles para controlar el acceso a los diferentes módulos del sistema
  - Módulo de Administración de parámetros del Sistema: Crear, eliminar, modificar variables que permitan parametrizar fácilmente aspectos generales del sistema, como lo son el acceso a la base de datos, repositorios para actualizaciones del sistema, direcciones de acceso al sistema para dispositivos móviles, etc.
  - Módulo de Gestión del Proceso de Recolección de Información Masiva con Tecnología Móvil: Cargar y procesar la información proporcionada y requerida por el cliente de ASISTECOM. Generar los reportes requeridos por ASISTECOM para control y seguimiento del proceso.
  
- Una Aplicación para dispositivos móviles, basados en Windows Mobile que permita:
  - Llevar a cabo el proceso de recolección de información in situ con dispositivos móviles.
  - Cargar y descargar de información desde y hacia los repositorios de información de la empresa.
    - Sincronización utilizando la interfaz USB de los equipos móviles.
    - Sincronización en línea, utilizando las redes de telefonía celular.

- Realizar las pruebas técnicas y funcionales de la aplicación, en base a las características antes descritas, previo la implantación en un ambiente de producción.

El nuevo sistema brindará todas las funcionalidades de la aplicación que se utiliza actualmente para la ejecución del proceso, permitirá gestionar no solo el proceso en sí, sino también los recursos utilizados (usuarios, equipos, etc.), optimizará la utilización de recursos tecnológicos con que cuenta la empresa y proporcionara soluciones a las deficiencias identificadas en el sistema actual.

Finalmente el nuevo sistema informático será desarrollado en una arquitectura distribuida, que brinde escalabilidad y permita la implementación de nuevos procesos y/o clientes no definidos como alcance de este proyecto.

#### **1.4. Objetivos**

##### **1.4.1. Objetivo General:**

Utilizar el Método Ágil SCRUM, aplicado a la Implantación de un Sistema Informático para el Proceso de RECOLECCIÓN MASIVA DE INFORMACIÓN IN SITU CON TECNOLOGÍA MÓVIL.

##### **1.4.2. Objetivos Específicos:**

- Revisar y analizar los conceptos relacionados a las metodologías de desarrollo de software.
- Analizar el método SCRUM y su aporte a la ejecución del presente proyecto.

- Analizar el proceso de Recolección de Información Masiva en la empresa ASISTECOM CÍA. LTDA.
- Diseñar la arquitectura a emplear en el Sistema Informático para la Administración del Proceso Operativo en Recolección de Información Masiva con Tecnología Móvil.
- Establecer SCRUM como el marco de trabajo para el desarrollo de la aplicación propuesta así como de futuras implementación de funcionalidades y/o mantenimiento de la misma.
- Desarrollo, pruebas e implantación del sistema Informático propuesto, basado en el método SCRUM.

## CAPÍTULO 2

### 2.1. Marco Teórico

Este capítulo tiene como objetivo, brindar una descripción del marco teórico de referencia al método ágil de desarrollo de software SCRUM previo a la ejecución de un ejercicio práctico, aplicado a la elaboración de un producto de software utilizado en el proceso de RECOLECCIÓN MASIVA DE INFORMACIÓN IN SITU CON TECNOLOGÍA MÓVIL.

En el ámbito de la ingeniería de software existe un continuo debate entre los partidarios de las metodologías tradicionales y aquellos que apoyan a las ideas emanadas del “Manifiesto Ágil”<sup>2</sup>.

Por un lado, para muchos equipos de desarrollo el uso de metodologías tradicionales resulta muy lejano a su forma de trabajo actual y la realidad de su cartera de proyectos, considerando las dificultades asociadas a la inversión en términos de formación y costos de herramientas utilizadas. Por otro lado están las características de los proyectos para los cuales las metodologías ágiles han sido especialmente pensadas; aquellos proyectos en los cuales los equipos de desarrollo son pequeños, de corto plazo, con requisitos volátiles y basados en nuevas tecnologías.

Las metodologías ágiles están especialmente orientadas a proyectos que necesitan soluciones a medida, con una elevada simplificación en términos de tiempo y recursos, sin dejar de lado el aseguramiento de la calidad del producto.

---

<sup>2</sup> (Independent Signatories of The Manifesto for Agile Software Development)



Estas metodologías se centran en el factor humano y el producto software; es decir, le dan mayor importancia al equipo de desarrollo, a la colaboración del cliente y al desarrollo incremental del software con iteraciones cortas (en tiempo).

## **2.2. Metodologías Tradicionales de Desarrollo de Software**

Inicialmente el desarrollo de software se lo llevaba a cabo de una forma totalmente artesanal. El crecimiento espectacular de la demanda de sistemas de computación cada vez más y más complejos, asociado a la inmadurez del propio sector informático y a la falta de métodos y recursos, provocó lo que se llamó la “crisis del software”<sup>3</sup> entre los años 1965 y 1985.

Durante esa época muchos proyectos importantes superaban ampliamente los presupuestos y fechas estimadas para su ejecución, por lo que las pérdidas iban mucho más allá del ámbito económico.

La importancia adquirida por los sistemas de computación dio lugar a una fuerte necesidad de mejorar el proceso para llevar los proyectos de desarrollo a la meta deseada.

Para resolver este problema fue necesario importar la concepción y fundamentos de metodologías existentes en otras áreas y adaptarlas al desarrollo de software. Esta adopción llevó a dividir el proceso de desarrollo en etapas generalmente secuenciales, lo que en algo ayudó en la latente necesidad de formalizar los procesos de desarrollo de software.

---

<sup>3</sup> F. L. Bauer, Primera conferencia de la tecnología de dotación lógica de la OTAN (Garmisch 1968)

Esta etapa de crisis fue superada no únicamente por la mejora en la gestión de los proyectos sino también por los progresos que se estaban dando en los procesos de diseño y metodologías de desarrollo.

La solución definitiva al problema de la planificación, previsión de costes y aseguramiento de la calidad en el proceso de desarrollo de software está ligada principalmente a la aparición de herramientas, metodologías y tecnologías.

Entre las principales metodologías de este tipo se encuentra RUP y MSF, que centran su atención en llevar una documentación exhaustiva de todo el proyecto y el cumplimiento estricto de un plan de proyecto, definido todo esto, en la fase inicial del proyecto.

Otras características importantes dentro de este enfoque son los altos costos de implementar un cambio y las dificultades de aplicarlo en proyectos donde el entorno es cambiante.

Las metodologías tradicionales (formales) se focalizan en la documentación, planificación y procesos. (Plantillas, técnicas de administración, revisiones, etc.).

## **2.2.1. Principales metodologías Tradicionales**

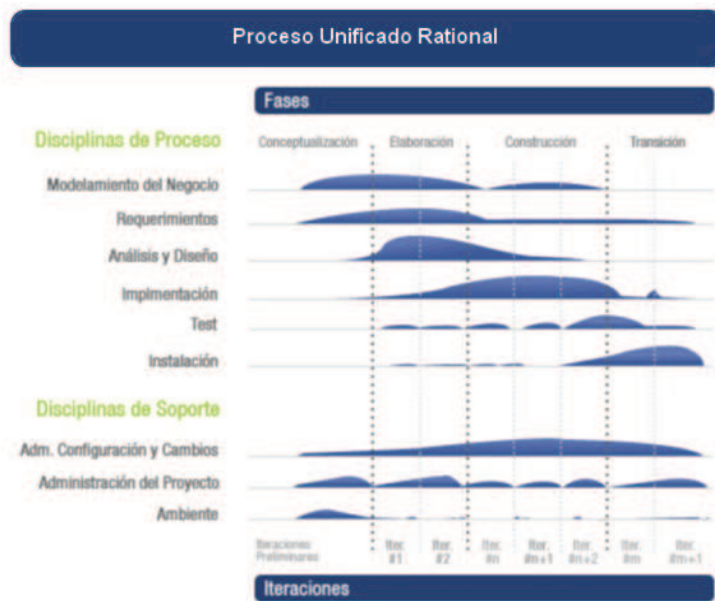
### **2.2.1.1. Rational Unified Process (RUP)<sup>4</sup>**

RUP provee un acercamiento disciplinado para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta calidad que satisfaga los

---

<sup>4</sup>RATIONAL, 31 de enero de 2012 <<http://www.rational.com.ar/herramientas/rup.html>>

requerimientos de los usuarios finales (respetando cronograma y presupuesto). Fue desarrollado por Rational Software, y está integrado con toda la suite de herramientas Rational. Puede ser adaptado y extendido para satisfacer las necesidades de la organización que lo adopte. Es guiado por casos de uso, centrado en la arquitectura y utiliza UML como lenguaje de notación.



**Figura 2. Rational Unified Process**

Fuente: Adaptado de (INFORMATICA ADSI , 2010)

### 2.2.1.2. MICROSOFT SOLUTION FRAMEWORK (MSF)<sup>5</sup>

MSF es un compendio de las mejores prácticas en cuanto a administración de proyectos se refiere. Más que una metodología rígida de administración de proyectos, MSF es una serie de modelos que puede adaptarse a cualquier proyecto de tecnología de información.

<sup>5</sup> MICROSOFT, 31 de enero de 2012 <<http://msdn.microsoft.com/es-es/library/ms195024%28v=vs.80%29.aspx>>

MSF divide a todo proyecto, separándolo en cinco fases principales:

- Visión y Alcances.
- Planificación.
- Desarrollo.
- Estabilización.
- Implantación.



Figura 3. MICROSOFT SOLUTION FRAMEWORK

Fuente: Adaptado de (Arevalo)

## 2.3. Metodologías Ágiles de Desarrollo de Software

### 2.3.1. Antecedentes

En febrero de 2001, tras una reunión celebrada en Utah-EEUU, aparece el término “ágil” aplicado al desarrollo de software. En esta reunión participaron un grupo de 17 expertos de la industria del software, incluyendo algunos de los creadores o impulsores de las metodologías de software existentes a la fecha.

Su objetivo fue perfilar los valores y principios que deberían permitir a los equipos de desarrollo, crear productos de software rápidamente y que estos equipos fueran capaces de asimilar rápidamente los cambios que puedan surgir a lo largo del proyecto. Se pretendía ofrecer una alternativa a los procesos de desarrollo de software tradicionales, que como se mencionó, estaban caracterizados por ser rígidos y dirigidos por la documentación que se genera en cada una de las actividades desarrolladas.

Tras esta reunión se creó The Agile Alliance<sup>3</sup>, una organización, sin fines de lucro, dedicada a promover los conceptos relacionados con el desarrollo ágil de software y ayudar a las organizaciones para que adopten dichos conceptos. El punto de partida fue el Manifiesto Ágil<sup>6</sup>, un documento que resume la filosofía “ágil”.

### **2.3.2. El manifiesto Ágil**

Los integrantes de la reunión resumieron los principios sobre los que se basan los métodos alternativos, en un documento denominado “Manifiesto Ágil”.

El manifiesto ágil está fundamentado en los siguientes valores.

- Al individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas. La gente es el principal factor de éxito de un proyecto de software. Es más importante construir un buen equipo que construir el entorno. Muchas veces se comete el error de construir primero el entorno y esperar que el equipo se adapte automáticamente. Es mejor crear el

---

<sup>6</sup> (Independent Signatories of The Manifesto for Agile Software Development)

equipo y que éste configure su propio entorno de desarrollo en base a sus necesidades.

- Desarrollar software que funciona más que conseguir una buena documentación. La regla a seguir es “no producir documentos a menos que sean necesarios de forma inmediata para tomar una decisión importante”. Estos documentos deben ser cortos y centrarse en lo fundamental.
- La colaboración con el cliente más que la negociación de un contrato. Se propone que exista una interacción constante entre el cliente y el equipo de desarrollo. Esta colaboración entre ambos será la que marque la marcha del proyecto y asegure su éxito.
- Responder a los cambios más que seguir estrictamente un plan. La habilidad de responder a los cambios que puedan surgir a lo largo del proyecto (cambios en los requisitos, en la tecnología, en el equipo, etc.) determina también el éxito o fracaso del mismo. Por lo tanto, la planificación no debe ser estricta sino flexible y abierta.

Los valores enumerados anteriormente inspiran los doce principios del manifiesto, estos son:

- I. La prioridad es satisfacer al cliente mediante tempranas y continuas entregas de software que le aporte un valor.
- II. Dar la bienvenida a los cambios. Se capturan los cambios para que el cliente tenga una ventaja competitiva.

- III. Entregar frecuentemente software que funcione desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre entregas.
- IV. La gente del negocio y los desarrolladores deben trabajar juntos a lo largo del proyecto.
- V. Construir el proyecto en torno a individuos motivados. Darles el entorno y el apoyo que necesitan y confiar en ellos para conseguir finalizar el trabajo.
- VI. El diálogo cara a cara es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo.
- VII. El software que funciona es la medida principal de progreso.
- VIII. Los procesos ágiles promueven un desarrollo sostenible. Los promotores, desarrolladores y usuarios deberían ser capaces de mantener una paz constante.
- IX. La atención continua a la calidad técnica y al buen diseño mejora la agilidad.
- X. La simplicidad es esencial.
- XI. Las mejores arquitecturas, requisitos y diseños surgen de los equipos organizados por sí mismos.
- XII. En intervalos regulares, el equipo reflexiona respecto a cómo llegar a ser más efectivo, y según esto ajusta su comportamiento.

### 2.3.3. Principales metodologías Ágiles

Aunque los creadores e impulsores de las metodologías ágiles más populares han suscrito el manifiesto ágil y coinciden con los principios enunciados anteriormente, existen una gran variedad de metodologías basadas en estos principios, cada una tiene características propias y enfatiza en algunos aspectos específicos. A continuación se resumen algunas metodologías ágiles. La mayoría de ellas han sido utilizadas con éxito en proyectos reales pero disponen de poca difusión o reconocimiento.

#### 2.3.3.1. SCRUM<sup>7</sup>

Método desarrollado por Ken Schwaber, Jeff Sutherland y Mike Beedle. Define un marco para la gestión de proyectos, que se ha utilizado con éxito durante los últimos 10 años.

Está especialmente indicada para proyectos con un rápido cambio de requisitos.



Figura 4. SCRUM

Fuente: Adaptado de (Epidata Consulting)

<sup>7</sup>CONTROLCHAOS ,31 de enero de 2012 <<http://www.scrum.org/scrumguides/>>



Sus principales características se pueden resumir en:

- El desarrollo de software se realiza mediante iteraciones, denominadas Sprints, con una duración variable de hasta 30 días. El resultado de cada sprint es un incremento ejecutable que se muestra al cliente.
- La segunda característica importante son las reuniones a lo largo del proyecto, entre ellas destaca la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración.

### 2.3.3.2. Extreme Programming<sup>8</sup>

La Programación Extrema (del inglés Extreme Programming) es uno de los llamados procesos o metodologías ágiles de desarrollo de software. Consiste en un conjunto de prácticas que a lo largo de los años han demostrado ser las mejores prácticas de desarrollo de software, llevadas al extremo, fundamentadas en los valores de las metodologías ágiles.



**Figura 5. Extreme Programming**

Fuente: Adaptado de (Sanchez, 2004)

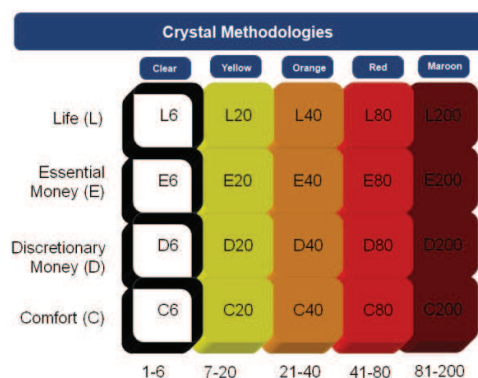
---

<sup>8</sup>PROGRAMACIONEXTREMA, 31 de enero de 2012 <<http://www.programacionextrema.org>>

### 2.3.3.3. Crystal Methodologies<sup>9</sup>

Se trata de un conjunto de metodologías para el desarrollo de software caracterizadas por estar centradas en las personas que componen el equipo y la reducción al máximo del número de artefactos producidos. Han sido desarrolladas por AlistairCockburn.

El desarrollo de software se considera un juego cooperativo de invención y comunicación, limitado por los recursos a utilizar. El equipo de desarrollo es un factor clave, por lo que se deben invertir esfuerzos en mejorar sus habilidades y destrezas, así como tener políticas de trabajo en equipo bien definidos. Estas políticas dependerán del tamaño del equipo, estableciéndose una clasificación por colores, por ejemplo Crystal Clear (3 a 8 miembros) y Crystal Orange (25 a 50 miembros).



**Figura 6. Crystal Methodologies**

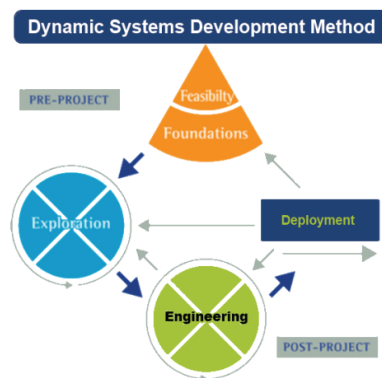
Fuente: Adaptado de (Wikiversity)

<sup>9</sup>CRYSTALMETHODOLOGIES, 31 de enero de 2012 <<http://www16.crystallmethodologies.org/>>

#### 2.3.3.4. Dynamic Systems Development Method (DSDM)<sup>10</sup>

Define el marco para desarrollar un proceso de producción de software. Nace en 1994 con el objetivo de crear una metodología RAD unificada. Sus principales características son:

- Es un proceso iterativo e incremental y el equipo de desarrollo y el usuario trabajan juntos.
- Propone cinco fases: estudio de viabilidad, estudio del negocio, modelado funcional, diseño y construcción, y finalmente implementación.
- Las tres últimas son iterativas, además de existir realimentación a todas las fases.



**Figura 7. Dynamic Systems Development Method**

Fuente: Adaptado de (Agile Certification)

<sup>10</sup>DSDM, 31 de enero de 2012 <[www.dsdm.org](http://www.dsdm.org)>

### 2.3.3.5. Feature -DrivenDevelopment (FDD)<sup>11</sup>

Define un proceso iterativo que consta de 5 pasos. Las iteraciones son cortas (hasta 2 semanas). Se centra en las fases de diseño e implementación del sistema partiendo de una lista de características que debe reunir el software. Sus impulsores son Jeff De Luca y Peter Coad.

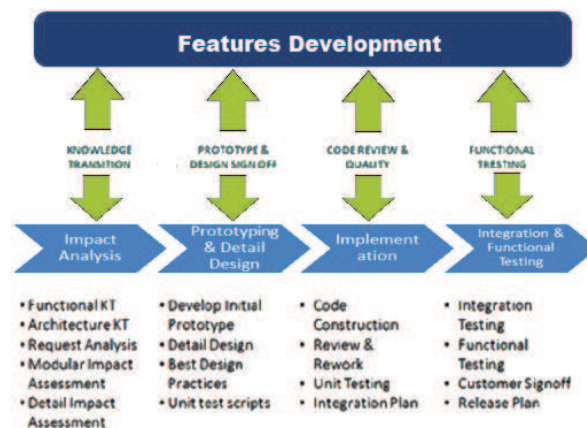


Figura 8. Feature -DrivenDevelopment

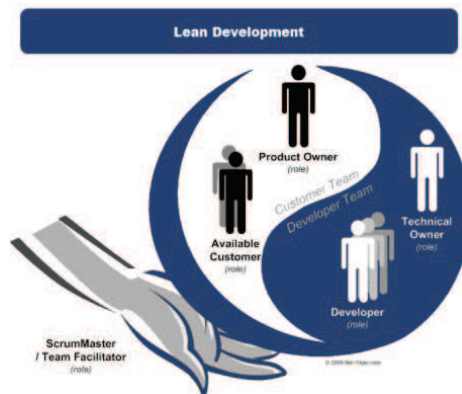
Fuente: Adaptado de (Texnovate Solutions Limited)

### 2.3.3.6. Lean Development(LD)<sup>12</sup>

Definida por Bob Charette.s a partir de su experiencia en proyectos con la industria japonesa del automóvil en los años 80 y utilizada en numerosos proyectos de telecomunicaciones en Europa. En LD, los cambios se consideran riesgos, pero si se manejan adecuadamente se pueden convertir en oportunidades que mejoren la productividad del cliente. Su principal característica es introducir un mecanismo para implementar dichos cambios.

<sup>11</sup> FEATUREDRIVENDEVELOPMENT,31 de enero de 2012<<http://www.featuredrivendevelopment.com/>>

<sup>12</sup>POPPENDIECK,31 de enero de 2012 <<http://www.poppendieck.com/>>



**Figura 9. Lean Development**

Fuente: Adaptado de (Net Objectives)

## 2.4. Metodologías Ágiles vs. Tradicionales

En el contexto del desarrollo de software, la necesidad de que los proyectos concluyan exitosamente y obtener un producto de gran valor para el cliente, se generan grandes cambios en las metodologías adoptadas por los equipos para cumplir sus objetivos, puesto que, unas se adaptan mejor que otras, al contexto del proyecto, brindando mejores ventajas.

El éxito del proyecto y la calidad del producto dependen en gran parte de la metodología escogida por el equipo, ya sea tradicional o ágil, donde los equipos maximicen su potencial, aumenten la calidad del producto con los recursos y tiempos establecidos.

El siguiente cuadro recoge las principales diferencias respecto del proceso, contexto de equipo y organización que es más favorable a cada una de estas filosofías de desarrollo de software.

**Tabla 3. Comparación entre metodologías ágiles y tradicionales**

<b>Metodologías Ágiles</b>	<b>Metodologías Tradicionales</b>
Basadas en heurísticas aplicadas en la prácticas de producción de código.	Basadas en estándares seguidos por el entorno de desarrollo.
Especialmente preparados para adaptarse a cambios durante el proyecto	Resistencia a los cambios imprevistos.
Impuestas internamente (por el equipo)	Impuestas externamente
Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas políticas/normas
No se rige a un contrato tradicional o al menos es bastante flexible	Se rige estrictamente a un contrato prefijado
El cliente interactúa con el equipo de desarrollo	El cliente es parte del equipo de desarrollo mediante reuniones
Grupos pequeños (10 integrantes máximo) y trabajando en el mismo sitio	Grupos grandes y posiblemente distribuidos
Pocos artefactos	Varios artefactos
Pocos roles	Varios roles
Menos énfasis en la arquitectura del software	La arquitectura del software es esencial y se expresa mediante modelos
Más utilizados en proyectos medianos y pequeños.	Más utilizado es proyectos grandes

Nota. Fuente: Adaptado de (Xelphos Group)

## **2.5. SCRUM**

### **2.5.1. Introducción**

SCRUM es un método ágil de gestión de proyectos cuyo objetivo principal es elevar al máximo la productividad de equipo de desarrollo. Reduce al máximo las actividades no orientadas a producir software funcional y produce resultados en periodos cortos de tiempo. Como método, enfatiza valores y prácticas de gestión, sin pronunciarse sobre requerimientos, prácticas de desarrollo, implementación y demás cuestiones técnicas. Más bien delega completamente al equipo la responsabilidad de decidir la mejor manera de trabajar para ser lo más productivos posibles.

La palabra SCRUM procede de la terminología del juego de rugby, donde SCRUM se define como el acto de preparar el avance del equipo en unidad pasando la pelota entre uno y otro jugador.

SCRUM fue desarrollado por Jeff Sutherland y elaborado más formalmente por Ken Schwaber. Poco después Sutherland y Schwaber se unieron para refinar y extender la metodología SCRUM y ha llegado a ser conocida como una herramienta de hiperproductividad. Schwaber se dio cuenta de que un proceso necesita aceptar el cambio, en lugar de esperar predictibilidad, esto enfocado en el hecho de que procesos definidos y repetibles sólo funcionan para atacar problemas definidos y repetibles con gente definida y repetible en ambientes definidos y repetibles. Tomar el cambio como una forma de entregar al final del desarrollo algo más cercano a la verdadera necesidad del Cliente fue entonces su solución.

SCRUM no se trata únicamente de un método para desarrollo de software, sino que puede ser aplicado teóricamente a cualquier contexto en donde un grupo de personas (equipo de trabajo) necesita trabajar junta para lograr una meta común.

Como metodología para desarrollo de software, se basa en los principios ágiles mencionados anteriormente, pero es necesario complementar el método SCRUM con otros métodos para su implementación en el ámbito de desarrollo de software.

### **2.5.2. La Esencia De SCRUM**

- Más que una metodología de desarrollo, SCRUM es una herramienta para gestionar proyectos.
- Está basado en el hecho de que el trabajo es realizado por equipos auto-organizado y auto-dirigidos, logrando motivación, responsabilidad y compromiso.
- Es un proceso constructivo, iterativo e incremental donde las iteraciones tienen una duración fija.
- Contiene definición de roles, prácticas y productos de trabajo escritas de forma simple y soportada en un conjunto de valores y principios (Métodos Ágiles).



### 2.5.3. Elementos del SCRUM

- Roles
  - Product Owner (Propietario del producto)
  - SCRUM Master
  - Team (Equipo)
- Poda de requerimientos
- Product Backlog
- Sprint
  - Planificación
  - Sprint Backlog
  - SCRUM
  - Estimaciones
  - Builds continuos
  - Revisión del Sprint
  - Reunión retrospective
- Valores
  - Foco, comunicación, respeto y coraje.

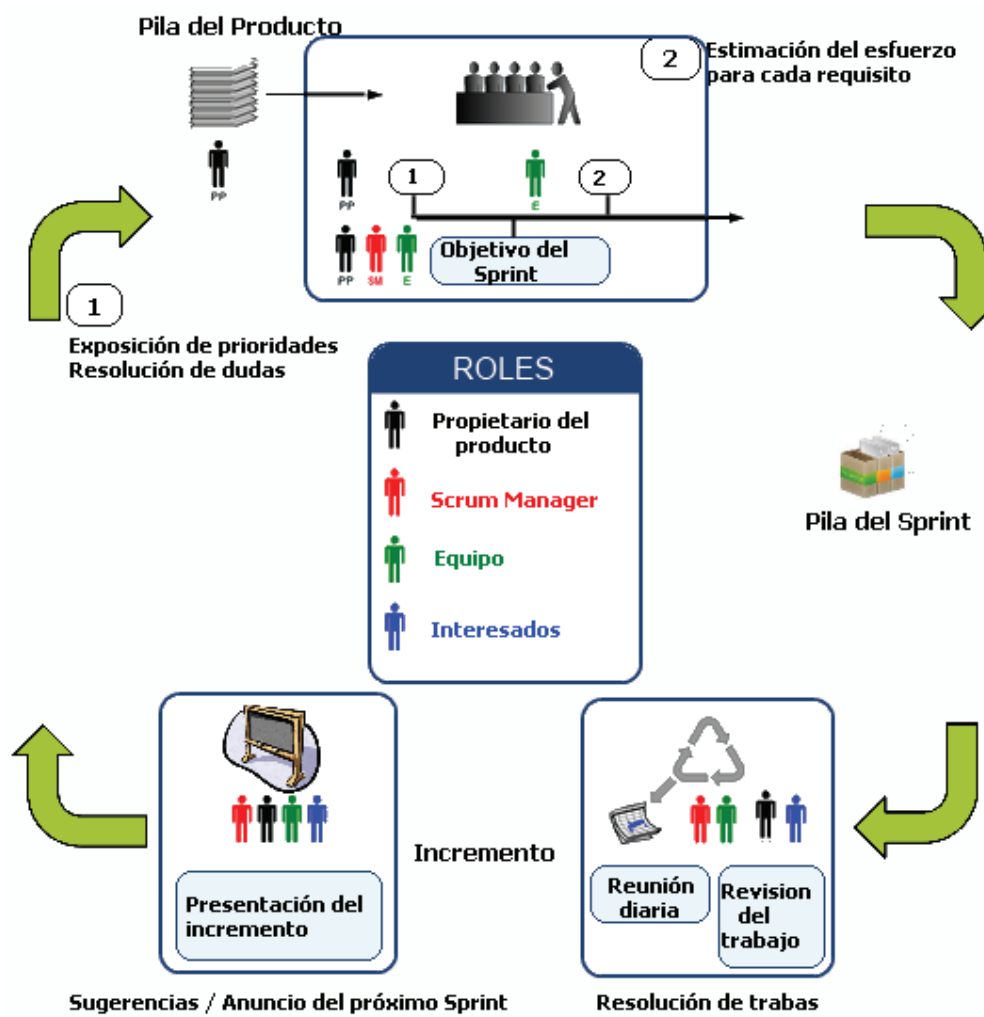


Figura 10. Elementos del SCRUM

Fuente: Adaptado de (Gestar)

### 2.5.3.1. Roles

El nombre de los miembros del equipo y los externos se deriva de una típica fábula agilista: un cerdo y una gallina discutían qué nombre ponerle a un nuevo restaurante; la gallina propuso “Jamón y Huevos”. “No, gracias”, respondió el cerdo, “Yo estaré comprometido, mientras tú sólo implicada”.

SCRUM diferencia claramente entre estos dos grupos para garantizar que quienes tienen la responsabilidad tienen también la autoridad necesaria para poder lograr el éxito, y que quienes no tienen la responsabilidad, los observadores externos, no produzcan interferencias innecesarias.

Tabla 4. Roles del SCRUM

Comprometido en el proyecto	Implicados en el proyecto
-Dueño del producto	-Marketing
-Equipo	-Comercial
-SCRUM Master	-Etc.

SCRUM tiene una estructura muy simple. Todas las responsabilidades del proyecto se reparten en 3 roles:

#### 2.5.3.1.1. Product owner (Dueño del producto)

Representa a todos los interesados en el producto final. Es el responsable oficial del proyecto, gestión, control y visibilidad de la lista de acumulación o lista de retraso del producto (Product Backlog). Toma

las decisiones finales de las tareas asignadas al registro y convierte sus elementos en rasgos a desarrollar.

Sus áreas de responsabilidad son:

- Financiación del proyecto
- Requisitos del sistema
- Retorno de la inversión del proyecto
- Lanzamiento del proyecto

#### **2.5.3.1.2. SCRUM Master (Líder del proyecto)**

Responsable del proceso SCRUM, de cumplir la meta y resolver los problemas. Así como también, de asegurarse que el proyecto se lleve a cabo de acuerdo con las prácticas, valores y reglas de SCRUM y que progrese según lo previsto.

Interactúa con el cliente y el equipo. Coordina los encuentros diarios, y se encarga de eliminar eventuales obstáculos. Debe ser miembro del equipo y trabajar a la par.

#### **2.5.3.1.3. Team (Equipo)**

Responsable de transformar el Backlog de la iteración en un incremento de la funcionalidad del software. Tiene autoridad para reorganizarse y definir las acciones necesarias o sugerir remoción de impedimentos.

La dimensión del equipo total de SCRUM no debería ser superior a veinte. El número ideal es diez, más o menos dos. Si hay más, lo más recomendable es formar varios equipos. No hay una técnica oficial para coordinar equipos múltiples, pero se han documentado experiencias de hasta 800 miembros, divididos en SCRUMs de SCRUMs, definiendo un equipo central que se encarga de la coordinación, las pruebas cruzadas y la rotación de los miembros.

### **2.5.3.2. Poda de requerimientos**

La poda de requerimientos es una buena práctica implícita en modelos ágiles, y consiste en hacer lo que el cliente realmente desea, no más.

La primera actividad en un proyecto manejado con SCRUM es armar una lista exhaustiva de los requerimientos originales del sistema. Posteriormente se realiza una revisión para evaluar qué requerimientos son realmente necesarios, cuáles pueden posponerse y cuáles eliminarse. Para ello debe identificarse un representante con capacidad de decisión, priorizar los requerimientos en base a su importancia y acordar cuáles son los prioritarios para la fecha de entrega.

### **2.5.3.3. Product Backlog**

Con los requerimientos priorizados y podados se arma el Backlog de Producto. Este es una forma de registrar y organizar el trabajo pendiente para el producto (actividades y requerimientos).

Es un documento dinámico que incorpora constantemente las necesidades del sistema. Por lo tanto, nunca llega a ser una lista completa y definitiva. Se mantiene durante todo el ciclo de vida y es responsabilidad del Product Owner.

#### **2.5.3.4. Sprint**

SCRUM está basado en el control empírico de procesos. Se utiliza cuando la capacidad de predicción es vaga, la incertidumbre alta o el proceso es demasiado complejo para ser modelado y definido.

En el enfoque empírico de control de procesos se establecen reglas simples y se crea una disciplina de inspección frecuente para adaptarse rápidamente a situaciones no previstas o problemas.

Un Sprint es el periodo de tiempo durante el que se desarrolla un incremento de funcionalidad. Constituye el núcleo de SCRUM, que divide de esta forma el desarrollo de un proyecto en un conjunto de pequeñas “carreras”.



**Figura 11. SPRINT**

Fuente: Adaptado de (Epidata Consulting)

Aspectos a tener en cuenta en un Sprint:

- Duración máxima del Sprint: 30 días, recomendada 15 días.
- Durante el Sprint no se puede modificar el trabajo que se ha acordado en el Backlog.
- Sólo es posible cambiar el curso de un Sprint, abortándolo, y sólo lo puede hacer el SCRUM Master si decide que no es viable por alguna de las razones siguientes:
  - La tecnología acordada no funciona.
  - Las circunstancias del negocio han cambiado.
  - El equipo ha tenido interferencias.

#### **2.5.3.4.1. Planificación**

Se planifica en detalle el trabajo al inicio de cada Sprint asumiendo que los objetivos no van a cambiar durante el mismo. De esta manera se atenúa el riesgo.

Aspectos a tener en cuenta sobre la planificación de un Sprint:

- Una determinación general de alcance, frecuentemente basada en una EDT (Estructura de División del Trabajo).
- Estimaciones de esfuerzo de alto nivel realizadas durante la etapa de concepción del proyecto.
- Esfuerzo dedicado a labores de soporte o de preparación de los ambientes requeridos por el proyecto.
- Esfuerzo asociados a las reuniones diarias, de planificación y de revisión.

- Requerimientos de recursos de infraestructura o logísticos (máquinas, redes, licencias, papel, pizarras, etc.).
- Habilidades presentes y necesarias en el equipo.
- Restricciones asociadas al conocimiento del negocio, la tecnología o externas (legales, reglamentarias, estándares, etc.).

El rol del SCRUM Master durante la planificación es:

- Dirigir la planificación.
- Facilitar acuerdos entre el equipo y el Product Owner del proyecto.
- Registrar problemas y riesgos detectados durante la planificación.
- Registrar las tareas, asignaciones y estimaciones.
- Iniciar el Backlog del Sprint.

#### **2.5.3.4.2. Sprint Backlog**

Trabajo o tareas determinadas por el equipo para realizar en un Sprint.

- Tareas a convertir en producto funcional.
- Las tareas se estiman en una duración entre 1 y 20 horas de trabajo. Las de mayor duración deben intentar descomponerse en sub-tareas en el rango de tiempo indicado.
- La estimación como el avance de las tareas se actualiza día a día.



#### **2.5.3.4.3. SCRUM diario**

SCRUM asume que el proceso es complejo y que es necesario inspeccionarlo frecuentemente, por eso se realiza una reunión diaria de seguimiento. El encuentro diario impide caer en el dilema señalado por Fred Brooks: “¿Cómo es que un proyecto puede atrasarse un año? Un día a la vez”.

El foco de la reunión es determinar el avance en las tareas y detectar problemas que estén haciendo lento el progreso del equipo o que eventualmente impidan a un equipo cumplir con la meta del Sprint. La idea es que ningún problema quede sin resolver o, por lo menos, sin iniciar alguna acción de respuesta dentro de las 24 horas después de su detección.

La reunión es además un espacio definido para que cada miembro del equipo comunique a los demás el estado de su trabajo.

El rol del SCRUM Master durante el SCRUM es:

- Dirigir la reunión y mantiene el foco.
- Hacer preguntas para aclarar dudas.
- Registrar (documentar) los problemas para su resolución después de la reunión.
- Asegura que los miembros cuenten con el ambiente adecuado para la reunión.

#### 2.5.3.4.4. Estimaciones

Las estimaciones se realizan por primera vez en la reunión de planificación al inicio del Sprint. Luego las tareas se re-estiman todos los días y se registran sus cambios en el Backlog del Sprint; esta actividad puede ser realizada inmediatamente antes o después del SCRUM diario.

#### 2.5.3.4.5. Builds continuos y pruebas básicas

La estrategia que generalmente se utiliza es la de Builds continuos y prueba básica para la funcionalidad del sistema consiste en:

- Los programadores desarrollan según el Backlog del Sprint, y al finalizar, notifican al integrador.
- El integrador toma el código y lo integra con el resto del producto.
- Se compila el software y se prueba superficialmente el producto, para verificar la estabilidad de la versión.
- Si se encuentran problemas se devuelve al desarrollador.
- Si no se han encontrado errores se notifica al equipo que hay una nueva versión “estable” del código para usar como base.

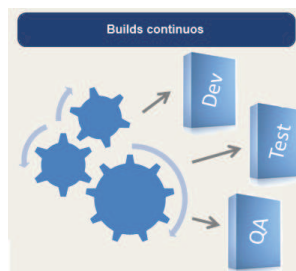


Figura 12. Builds continuos

#### **2.5.3.4.6. Revisión del Sprint**

El objetivo de la reunión de revisión es presentar el producto o porción del producto desarrollada por el equipo a los usuarios. La reunión se utiliza para detectar inconformidades mayores que se vuelven elementos del Backlog de Producto y que eventualmente se resuelven en el siguiente Sprint.

A la reunión asisten el equipo, el SCRUM Master, el Product Owner y todas las personas implicadas en el proyecto. Esta revisión se convierte en un demo del producto con las nuevas funcionalidades desarrolladas durante el Sprint.

#### **2.5.3.4.7. Reunión retrospectiva**

SCRUM involucra el concepto de mejora continua a través de las reuniones de retrospección. Las reuniones buscan detectar los puntos positivos y negativos del Sprint para generar propuestas de mejora para futuros Sprints.

Las reuniones de retrospección son el concentrador del aprendizaje organizacional sobre el SCRUM. Los puntos positivos y negativos se registran y se definen ítems de acción para cada uno. Los ítems de acción definidos se toman en cuenta en los siguientes Sprints.

A esta reunión acuden el equipo, el SCRUM Master, y opcionalmente el Product Owner del Producto.

## **2.5.3.5. Valores**

### **2.5.3.5.1. Foco**

Los individuos y sus interacciones son más importantes que el proceso o las herramientas. El recurso humano es el principal factor de éxito de un proyecto de software.

### **2.5.3.5.2. Comunicación**

SCRUM pone en comunicación directa y continua a clientes y desarrolladores. El cliente se integra en el equipo para establecer prioridades y resolver dudas. De esta forma ve el avance día a día, y es posible ajustar la agenda y las funcionalidades de forma consecuente.

### **2.5.3.5.3. Respeto**

SCRUM diferencia claramente entre dos grupos para garantizar que quienes tienen la responsabilidad tienen también la autoridad necesaria para poder lograr el éxito (cerdos), y que quienes no tienen la responsabilidad, los observadores externos (gallinas), no produzca interferencias innecesarias.

### **2.5.3.5.4. Coraje**

El coraje implica saber tomar decisiones difíciles. Reparar un error cuando se detecta. Mejorar el código siempre que el feedback y las sucesivas iteraciones se manifiesten susceptibles de mejora.

## CAPÍTULO 3

### 3.1. Metodología

Se entiende por metodología a la colección de documentación formal referente a los procesos, políticas y procedimientos que intervienen en las diferentes etapas de la ejecución de un proceso.<sup>13</sup>

Una metodología de desarrollo de software se refiere a un marco de trabajo utilizado para estructurar, planear y controlar el proceso de desarrollo en sistemas de información. Este marco de trabajo debe ser enfocado al proceso de desarrollo de software con la utilización de herramientas, modelos y métodos que garanticen un producto de calidad.

La finalidad de una metodología de desarrollo es garantizar la eficacia (cumplir los requisitos) y la eficiencia (optimizar recursos) en el proceso de generación de software.

Una metodología puede seguir uno o varios modelos de ciclo de vida del software que indican qué es lo que se obtendrá a lo largo del desarrollo del proyecto, durante este desarrollo, la metodología indica cómo hay que obtener los distintos productos parciales y finales.

La esencia de cualquier metodología de desarrollo es la elaboración de documentos escritos que detallan cada etapa del ciclo de vida de desarrollo.

---

<sup>13</sup> Blanco, S. (s.f.). *Marble Station*. Recuperado el 15 de 03 de 2012, de <http://www.marblestation.com/?p=644>

### **3.2. Desarrollo iterativo e incremental**

En un desarrollo iterativo e incremental un proyecto de desarrollo se planifica en diversos bloques temporales, en el caso de SCRUM estos periodos de tiempo, denominados iteraciones o Sprints tienen una duración recomendada de entre dos a cuatro semanas.

Cada iteración se puede entender como un pequeño proyecto individual: en cada iteración se repite un proceso de trabajo similar "iterativo" para proporcionar un resultado completo sobre el producto final, para que el cliente pueda obtener los beneficios del proyecto de forma incremental. Para ello, es recomendable que cada requisito se complete en una única iteración: el equipo debe realizar todas las tareas necesarias para completarlo (incluyendo pruebas y documentación) y que esté preparado para ser entregado al cliente con el mínimo esfuerzo necesario. De esta manera no se deja para el final del proyecto ninguna actividad relacionada con la entrega de requisitos.

Para la ejecución del presente proyecto, se ha propuesto un proceso iterativo e incremental. Iterativo para garantizar la realimentación de información y de requisitos una vez iniciado el desarrollo, así como la validación continua del sistema. Esto permite que cada iteración contemple ciclos de desarrollo completos, cortos y que permitan obtener rápidamente una versión de software con mejoras sobre las versiones anteriores. Incremental en el sentido de añadir capacidades y funcionalidades al software de acuerdo a la planificación inicial; con el propósito de obtener el sistema final tras la realización de diferentes ciclos. Este proceso permite realizar entregas al cliente de forma rápida y ágil.

La estrategia de desarrollo propuesta permite a la vez enfocarse en las necesidades de los usuarios, involucrándolos con las entregas de versiones estables e instándolos a identificar sus prioridades en cada etapa del proyecto.

### **3.3. Etapas Del Proceso De Desarrollo**

Las etapas de desarrollo de software son un reflejo del proceso que se sigue a la hora de resolver cualquier tipo de problema. Ya en 1945, mucho antes de que existiese la Ingeniería del Software, el matemático George Polya describió este proceso en su libro "How to solve it" seguramente el primero que describe la utilización de técnicas heurísticas en la resolución de problemas. Básicamente, resolver un problema requiere:

- Comprender el problema (análisis).
- Plantear una posible solución, considerando soluciones alternativas (diseño).
- Llevar a cabo la solución planteada (implementación).
- Comprobar que el resultado obtenido es correcto (pruebas).

En este sentido, cualquier sistema de información pasa por una serie de fases a lo largo de su vida. Su ciclo de vida comprende una serie de etapas entre las que se encuentran las siguientes:

- Planificación
- Análisis

- Diseño
- Implementación y Pruebas
- Instalación o despliegue
- Uso y mantenimiento

Para cada una de las fases del ciclo de vida de un del desarrollo de software se han propuesto prácticas útiles, entendiendo por prácticas aquellos conceptos, principios, métodos y herramientas que facilitan la consecución de los objetivos de cada etapa.

### 3.3.1. Planificación

Antes de iniciar un proyecto de desarrollo de software, es necesario realizar una serie de tareas previas, las mismas que influirán decisivamente en la finalización exitosa del proyecto. Estas tareas normalmente no están sujetas a plazos. Las tareas que se realizarán esta fase inicial del proyecto incluyen actividades tales como la determinación del ámbito del proyecto, la realización de un estudio de viabilidad, una estimación del coste del proyecto, su planificación temporal y la asignación de recursos a las distintas etapas del proyecto .

- **Objetivo:** Definir el proyecto propiamente dicho.
- **Tareas:** Relevamiento preliminar de los procesos del negocio y definición de actividades, definición del alcance, estimación de tiempos, definición de recursos, estimación de costos.



- **Entregables:** Documento de definición del proyecto o del Sprint.

### 3.3.2. Análisis

Antes de iniciar la etapa de construcción es necesario conocer exactamente lo que tiene que hacer el sistema. La etapa de análisis en el ciclo de vida del software corresponde al proceso mediante el cual se intenta descubrir qué es lo que realmente se necesita y se llega a una comprensión adecuada de los requerimientos del sistema (las características que el sistema debe poseer).

- **Objetivo:** Obtener todas las definiciones y especificaciones funcionales para poder llevar adelante las fases de Diseño y Construcción. Es una etapa clave ya que el alcance y las características de la solución quedan acordados, lo cual permite mitigar los principales riesgos de un proyecto.
- **Tareas:** Afianzamiento de las definiciones funcionales, definición de los requisitos a través de casos de uso, planificación de las etapas posteriores y ajuste de los tiempos preestablecidos.
- **Entregable:** Documento de alcance, casos de uso y sus respectivas descripciones.

### 3.3.3. Diseño

Mientras que los modelos utilizados en la etapa de análisis representan los requisitos del usuario desde distintos puntos de vista (¿Qué hacer?), los modelos que se utilizan en la fase de diseño representan las características del sistema que permitirán implementarlo de forma efectiva (¿Cómo Hacerlo?).

Un software bien diseñado debe exhibir determinadas características. Su diseño debería ser modular en vez de monolítico. Sus módulos deberían ser cohesivos (encargarse de una tarea concreta y sólo de una) y estar débilmente acoplados entre sí (para facilitar el mantenimiento del sistema). Cada módulo debería ofrecer a los demás unos interfaces bien definidos. Por último, debe ser posible relacionar las decisiones de diseño tomadas con los requerimientos del sistema que las ocasionaron.

- **Objetivo:** Generar el modelo de datos para que la solución cumpla con los requerimientos definidos. El diseño generado deberá contemplar las posibles modificaciones futuras, crecimiento de la solución, mayor carga e incorporación de nuevas funcionalidades.
- **Tareas:** Diagrama Entidad Relación (DER), diseño de las interfaces de usuario, diseño de las integraciones a realizar. Durante esta etapa también se realizan pruebas para puntos críticos del proyecto.
- **Entregables:** Entre los entregables típicos de esta etapa se encuentran: DER, esqueleto del software armado, guía de diseño, diseño de la infraestructura, y la planificación ajustada con la evolución y avances obtenidos.

#### **3.3.4. Construcción y Pruebas**

Una vez identificadas las funciones que debe desempeñar el sistema de información (análisis) y decidido cómo organizar sus distintos componentes (diseño), es momento de pasar a la etapa de implementación. Antes de iniciar la codificación o crear una tabla en nuestra base de datos es fundamental haber

comprendido bien el problema que se pretende resolver y haber aplicado principios básicos de diseño que permitan construir un sistema de información de calidad.

Para la fase de implementación es necesario seleccionar las herramientas adecuadas, un entorno de desarrollo que facilite el trabajo y un lenguaje de programación apropiado para el tipo de sistema a construir. La elección de estas herramientas dependerá en gran parte de las decisiones de diseño que se han tomado hasta el momento y del entorno en el que el sistema deberá funcionar.

Durante la etapa de construcción se puede iniciar también la etapa de pruebas que tiene como objetivo detectar los errores que se hayan podido cometer tanto durante la codificación como en las demás etapas anteriores. El propósito, es hacerlo antes de que el usuario final del sistema realice sus pruebas de funcionamiento.

La etapa de pruebas puede adoptar distintas formas, en función del contexto y de la fase en que se encuentre el proyecto, estas pueden ser:

- **Pruebas de unidad:** sirven para comprobar el correcto funcionamiento de un componente concreto de nuestro sistema. Es este tipo de pruebas, buscar situaciones que expongan las limitaciones de la implementación del componente, en estas pruebas el componente puede ser tratado como una caja negra ("pruebas de caja negra") o con un análisis de su estructura interna ("pruebas de caja blanca"). Resulta recomendable que, conforme se añaden nuevas funcionalidades a la aplicación, se

creen nuevos tests con los cuales medir el progreso; también es importante repetir los test antiguos para comprobar que lo que antes funcionaba sigue haciéndolo.

- **Pruebas de integración:** son las que se realizan cuando se van juntando los componentes que conforman el sistema y sirven para detectar errores en sus interfaces.
  - **Pruebas Alfa:** Una vez "finalizado" el sistema, los responsables del desarrollo realizan pruebas desde el punto de vista de un usuario final, este tipo de pruebas pueden ayudar principalmente a pulir aspectos de la interfaz de usuario del sistema y validaciones que pudieron haberse pasado por alto durante la codificación.
  - **Pruebas beta:** Cuando el sistema no es un producto a medida, sino que se venderá como un producto en el mercado, se deben realizar estas pruebas, que serán ejecutadas por usuarios finales del sistema (clientes potenciales) ajenos al equipo de desarrollo. En el caso del presente proyecto, este tipo de pruebas no aplica.
  - **Pruebas de Aceptación:** En sistemas a medida, estas pruebas son realizadas por los usuarios finales y si se supera con éxito, marcará oficialmente el final del proceso de desarrollo.
- **Objetivo:** Construir la solución del Release (Sprint), cumpliendo con las definiciones y especificaciones de los documentos de alcance.

Generalmente es la etapa de mayor duración y con mayor dinámica de trabajo.

- **Tareas:** Programación y desarrollo de todos los componentes que brinden las funcionalidades requeridas. Implementación de las estructuras de datos, y sus procedimientos, elaboración de documentación técnica y ajustes funcionales, implementación de las integraciones y todas las actividades necesarias para poner en marcha la solución. En esta etapa se realizarán las pruebas descritas anteriormente.
- **Entregables:** El entregable principal es el incremento de software funcionando.

### **Pruebas de software basado en Modelos**

Las pruebas basadas en modelos son definidas como las pruebas de software donde los casos de pruebas son derivados de un modelo que describe algunos o todos los aspectos del sistema a probar. El sistema a probar puede ser tan simple como un método o una clase, o tan complejo como un sistema completo. Para este tipo de pruebas, un modelo proporciona una descripción del comportamiento del sistema puesto a prueba. Esta descripción es utilizada para determinar si el sistema bajo prueba se ajusta las propiedades descritas en el modelo.

Existen varios modelos utilizados para el proceso de pruebas, de los cuales se puede destacar los ilustrados en las figuras 13, 14 y 15.

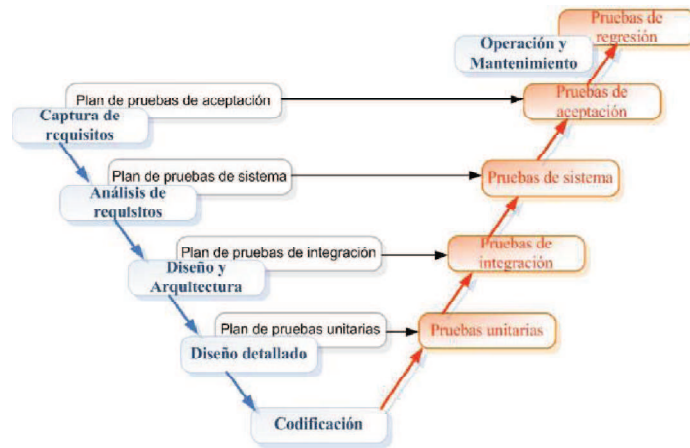


Figura 13. Modelo V 14

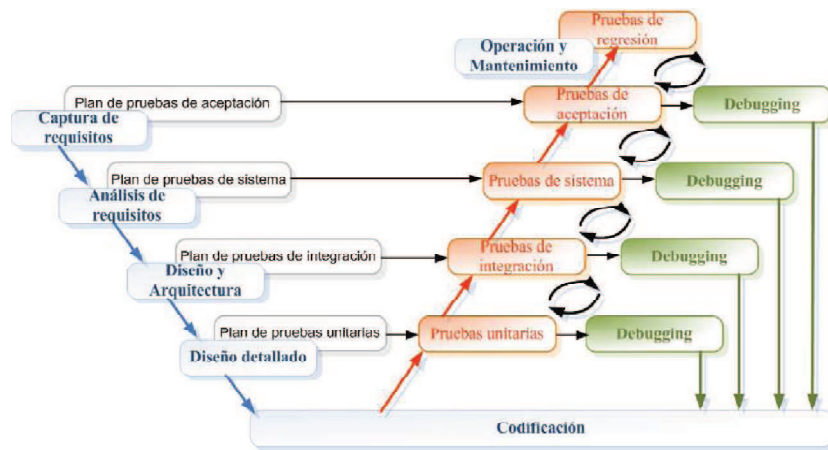
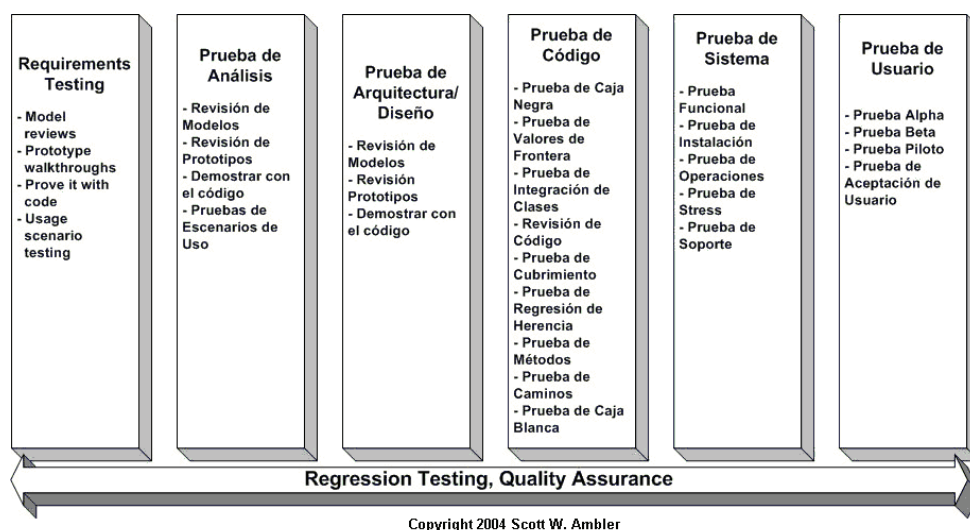


Figura 14. El modelo W<sup>15</sup>

<sup>14</sup> ( Zenteno, A. H. (2008). *Método de pruebas de sistema basado en modelos navegacionales en un contexto MDWE*. Recuperado el 15 de 03 de 2012, de [www.lsi.us.es/docs/doctorado/.../MemolInvestigArturoHTorres.pdf](http://www.lsi.us.es/docs/doctorado/.../MemolInvestigArturoHTorres.pdf)

<sup>15</sup> Zenteno, A. H. (2008). *Método de pruebas de sistema basado en modelos navegacionales en un contexto MDWE*. Recuperado el 15 de 03 de 2012, de [www.lsi.us.es/docs/doctorado/.../MemolInvestigArturoHTorres.pdf](http://www.lsi.us.es/docs/doctorado/.../MemolInvestigArturoHTorres.pdf)



**Figura 15. Modelo de Pruebas Orientada a Objetos para el Ciclo de Vida Completo.**<sup>16</sup>

El Modelo de la Figura 15, es un modelo muy completo, que se enfoca no únicamente en la etapa de construcción, sino que, analiza todas las etapas de ciclo de vida del software, es por ello que se ha decidido utilizar el mencionado modelo; sin embargo, para el presente proyecto las pruebas serán enfocaran únicamente para las fases de de codificación, pruebas del sistema y pruebas de usuarios. Los modelos utilizados en la etapa de análisis han sido probados y se encuentran actualmente implementados en los procesos operativos de ASISTECOM, por tanto no serán sometidos a pruebas. Así mismo los modelos utilizados en la etapa de diseño, son modelos estándar, por lo que no necesitan ser analizados ni probados nuevamente.

Las técnicas de pruebas utilizadas en el modelo seleccionado son las descritas en la Tabla 5. Técnicas de pruebas.

<sup>16</sup> (Ambler, 2004)

Las pruebas utilizadas dependerán de las funcionalidades de componente a probar y estas pueden variar entre los distintos módulos del sistema;

**Tabla 5. Técnicas de pruebas**

<b>Técnica</b>	<b>Descripción</b>
Prueba de Caja-Negra	Centrada en un estudio desde el punto de vista de las entradas que recibe y las salidas que produce, sin tener en cuenta su funcionamiento interno.
Prueba de Valores-Frontera	Centrada en probar situaciones extremas o inusuales que el sistema debe ser capaz de manejar.
Prueba de Clases	Es el acto de asegurar que una clase y todas sus instancias cumplen con el comportamiento definido.
Prueba de Integración de Clases	Es el acto de asegurar que las clases, y sus instancias, conforman un software que cumple con el comportamiento definido.
Revisión de Código	Una forma de revisión técnica en la que el entregable que se revisa en el código fuente.
Prueba de Componente	Es el acto de validar que un componente funciona tal como está definido.
Prueba de Cubrimiento	Es el acto de asegurar que toda línea de código es ejercita al menos una vez.
Revisión de Diseño	Una revisión técnica en la cual se inspecciona un modelo de diseño.
Prueba de Regresión de Herencia	Es el acto de ejecutar casos de prueba de las súper clases, tanto de forma directa como indirecta, en una subclase específica.
Prueba de Integración	Consiste en realizar pruebas para verificar que un gran conjunto de partes del software funcionan juntas.
Prueba de Método	Consiste en realizar pruebas para verificar que un método (función miembro) funciona tal como está definido.
Revisión de Modelos	Un tipo de inspección, que puede ser desde un revisión técnica formal hasta un recorrido informal, realizado por personas diferentes a las que estuvieron directamente involucradas en el desarrollo del modelo.
Prueba de Caminos	Es el acto de asegurar que todos los caminos lógicos en el código se ejercitan al menos una vez.



Técnica	Descripción
Revisión de Prototipos	Es un proceso mediante el cual los usuarios trabajan a través de una colección de casos de uso, utilizando un prototipo como si fuera el sistema real. El objetivo principal es probar si el diseño del prototipo satisface las necesidades de esos usuarios.
Demostrar con el código	La mejor forma de determinar si un modelo realmente refleja lo que se necesita, o lo que se debe construir, es construyendo software basado en el modelo para mostrar que el modelo está bien
Prueba de Regresión	El acto de asegurar que los comportamientos previamente probados todavía trabajan como se espera luego que se han realizado cambios a la aplicación.
Prueba de Stress	El acto de asegurar que el sistema funciona como se espera bajo grandes volúmenes de transacciones, usuarios, carga y demás.
Revisión Técnica	Una técnica de aseguramiento de la calidad en la cual el diseño de tu aplicación es revisado de forma exhaustiva por un grupo de tus compañeros. Una revisión típicamente se enfoca en la precisión, calidad, facilidad de uso y completitud. A este proceso usualmente se le llama recorrido, inspección, o revisión de compañeros.
Prueba de Escenarios de Uso	Una técnica de prueba en la cual una o más personas validan un modelo siguiendo la lógica de los escenarios de uso.
Prueba de Interfaz de Usuario	Consiste en probar la interfaz de usuario para garantizar que cumple los estándares y requerimientos definidos. Usualmente se refiere a la prueba de interfaz de usuario gráfica.
Prueba de Caja-Blanca	Centrada en los detalles procedimentales del software, por lo que está fuertemente ligado al código fuente.

**Nota:** Las técnicas utilizadas para probar el presente sistema dependerán del módulo y los componentes a probar. <sup>17</sup>

<sup>17</sup> Ambler, S. W. (04 de 2004). *Ambysoft*. Recuperado el 06 de 04 de 2012, de <http://www.ambyssoft.com/essays/flootSpanish.html>

### 3.3.5. Implementación

Una vez concluidas las etapas de desarrollo de un sistema de información (análisis, diseño, implementación y pruebas), llega el momento de poner el sistema en funcionamiento, su instalación o despliegue.

Antes de la instalación propiamente dicha, es necesario planificar el entorno en el que el sistema debe funcionar, tanto a nivel de hardware como a nivel de software: equipos necesarios y su configuración física, redes de comunicación entre los equipos y de acceso a sistemas externos de ser necesario, sistemas operativos (actualizados para evitar problemas de seguridad), bibliotecas y componentes suministrados por terceros, etc.

- **Objetivo:** Disponer del sistema productivo en un ambiente de producción, metodología de trabajo y manuales operativos. Se incluye, de ser necesario, el personal operativo capacitado. Obtención de nuevas funciones a agregar o posibles errores a reparar.
- **Tareas:** Puesta en marcha de la aplicación en el ambiente de producción, elaboración de manuales operativos, y todas las actividades relacionadas al éxito del lanzamiento como la integración del ambiente de producción con terceras partes, etc.
- **Entregables:** El sistema productivo con sus manuales operativos, de mantenimiento y de procedimientos. Sistema totalmente probado.

### **3.4. Herramientas**

#### **3.4.1. Técnicas de relevamiento**

Para la ejecución del presente proyecto se utilizará la observación, las entrevistas con el cliente y los usuarios como principal técnica de relevamiento.

#### **3.4.2. Casos de Uso**

Son un método práctico para explorar requerimientos. Ayudan a describir qué es lo que el sistema debe hacer desde el punto de vista del usuario. Por lo tanto, se considera que los casos de uso proporcionan un modo claro y preciso de comunicación con el cliente. Los diagramas de casos de uso describen las relaciones y las dependencias entre un grupo de casos de uso y los actores participantes en el proceso.

Es importante resaltar que los diagramas de casos de uso no están pensados para representar el diseño y no puede describir los elementos internos de un sistema. Los diagramas de casos de uso sirven para facilitar la comunicación con los futuros usuarios del sistema, y con el cliente, y resultan especialmente útiles para determinar las características necesarias que tendrá el sistema. Esto es, los diagramas de casos de uso describen qué es lo que debe hacer el sistema, pero no cómo.

Una vez realizados los diagramas de casos de uso, que describen gráficamente lo que debe hacer el sistema, se detallan los casos de uso, en ellas se explica la forma de interactuar entre el sistema y el usuario.

Los casos de uso son utilizados durante la etapa de análisis.

Para la ejecución del presente proyecto se utilizará el formato de especificación de requerimientos presentado en la Figura 16.

<b>Nombre</b>	
<b>ID</b>	<Identificados del caso de uso en el diagrama>
<b>Descripción</b>	<Descripción del caso de uso>
<b>Precondición</b>	<Condiciones que deben presentarse antes de llegar al caso de uso>
<b>Post condición</b>	<Condiciones generadas por el caso de uso>
<b>Flujo Normal</b>	<Descripción del flujo normal del caso de uso>
<b>Flujos Alternos</b>	<En caso de que el caso de uso puede tener flujos alternos, describirlos en esta parte>
<b>Excepciones</b>	<Las excepciones a los flujos normales también son importantes>
<b>Notas:</b>	<Otros datos que aporten a la implementación.>

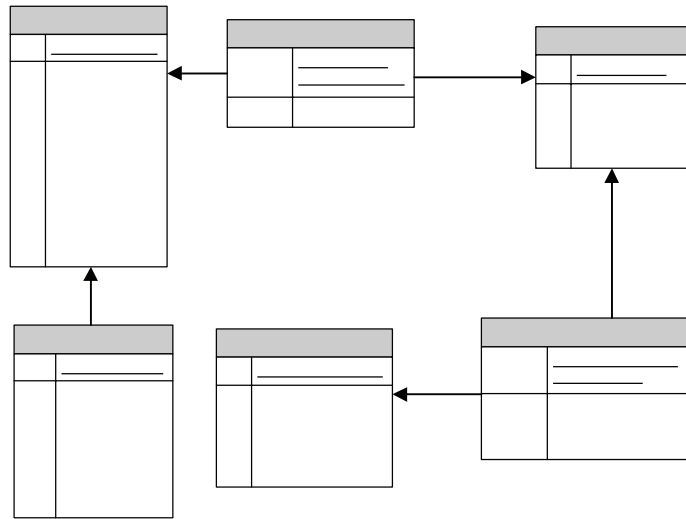
Figura 16. Plantilla para especificación de casos de uso.

### 3.4.3. Diagrama de Entidad Relación (DER)

Un modelo de datos describe de una forma abstracta cómo se representan los datos, sea en una empresa, en un sistema de información o en un sistema de base de datos.

Como herramienta para el modelado de datos de un sistema de información, los DER expresan entidades relevantes y sus inter-relaciones. Formalmente, son un lenguaje gráfico para describir conceptos y describen la información utilizada en un sistema de información.

El modelo de datos entidad-relación está basado en una percepción del mundo real que consta de una colección de objetos básicos, llamados entidades, y de relaciones entre esos objetos.



**Figura 17. Ejemplo Diagrama Entidad Relación**

#### 3.4.4. **Sprintometer**

Permite llevar a cabo el seguimiento del proyecto. Es una herramienta de automatización de procesos ágiles que admite a los equipos auto organizarse y aumentar la productividad. Esta herramienta, de acceso libre y fácil de utilizar, es una aplicación de escritorio que permite alojar en la nube de forma gratuita los datos referentes al proyecto y compartir la información del proyecto entre todo el equipo.

Esta herramienta para la administración del proyecto permite:

- Manejar proyectos y definir para cada proyecto los Sprint, las historias de usuario (Story) y las respectivas tareas asociadas a cada historia.
- Actualizar continuamente los avances realizados en las tareas asignadas.
- Observar un gráfico por cada Sprint que indica la velocidad con la que avanza el proyecto. Estos gráficos llamados “burndown” no sólo permiten

observar el estado de avance del proyecto, sino también analizar sus comportamientos e ir aprendiendo para mejorar los Sprints que restan.

Esta herramienta será utilizada en todas las etapas de cada Sprint.

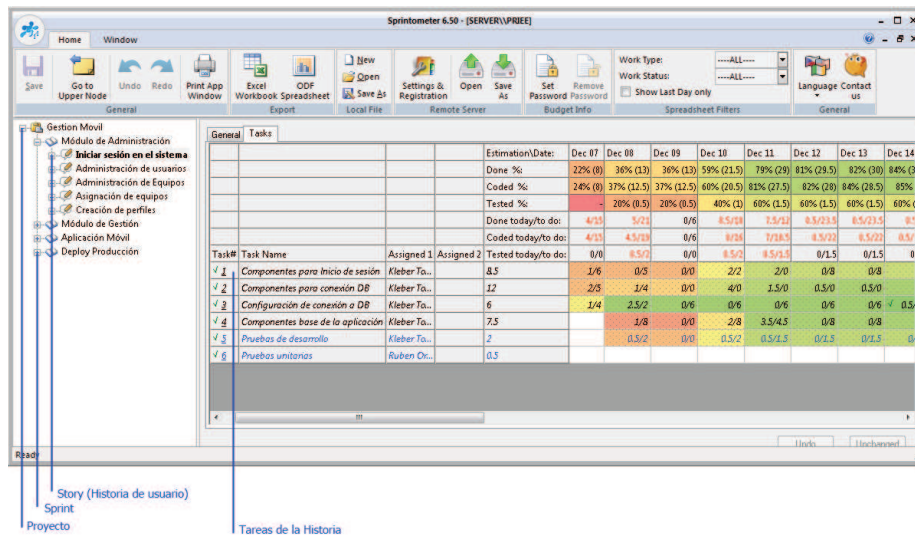


Figura 18. Definiciones Sprintometer

### 3.4.5. Visual Studio 2010

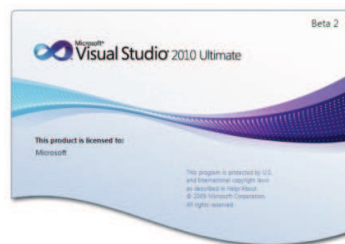
Microsoft Visual Studio es un entorno de desarrollo integrado para sistemas operativos Windows. Soporta varios lenguajes de programación tales como Visual C++, Visual C#, Visual J#, ASP.NET y Visual Basic .NET, aunque actualmente se han desarrollado las extensiones necesarias para muchos otros lenguajes como python, ruby, etc.

Visual Studio permite a los desarrolladores crear aplicaciones, sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET (a partir de la versión .NET 2002). Así se pueden crear

aplicaciones que se intercomunican entre estaciones de trabajo, páginas web y dispositivos móviles.

Estas características han hecho que se elija Visual Studio 2010 como la herramienta de desarrollo para codificar el presente proyecto.

Esta herramienta es utilizada durante la etapa de construcción.



**Figura 19. Herramienta de desarrollo VS2010.**

#### **3.4.6. Jira**

JIRA es una herramienta de gestión de proyectos en línea que ayuda a los equipos a construir mejor software gestionando principalmente bugs, tareas permitiendo un continuo monitoreo de actividades, y obtener informes sobre el estado de las incidencias o bugs levantada, especialmente en la etapa de pruebas. De esta manera tanto el equipo de desarrollo como los dueños del producto, pueden conocer el estado de los errores reportados.

Este tipo de herramientas son de gran utilidad cuando el equipo de trabajo no se encuentra físicamente juntos, lo cual no implica que deban permanecer desinformado del estado del proyecto y en este caso particular de los errores reportados.

## **CAPÍTULO 4**

### **4.1. Ejecución del Proyecto**

#### **4.1.1. Planificación**

Como se había mencionado en el capítulo anterior, la primera fase de del proceso de desarrollo es la planificación; esta tapa al igual que el resto de etapas serán analizadas una vez durante el desarrollo de cada Sprint. Sin embargo, antes de iniciar con las iteraciones, es necesario realizar una definición macro de las iteraciones, esto es, realizar una planificación inicial, que permita identificar el propósito de cada iteración y definir en forma global lo que se realizará en cada sprint. Este análisis inicial se denomina “Sprint 0”.

Antes de definir el número y los objetivos de cada Sprint a desarrollar durante la ejecución del presente proyecto, es necesario definir ciertas actividades, una de ellas, el análisis preliminar de los procesos del negocio, definir los alcances del presente proyecto, realizar una estimación de tiempos y recursos a utilizar.

##### **4.1.1.1. Análisis de Procesos**

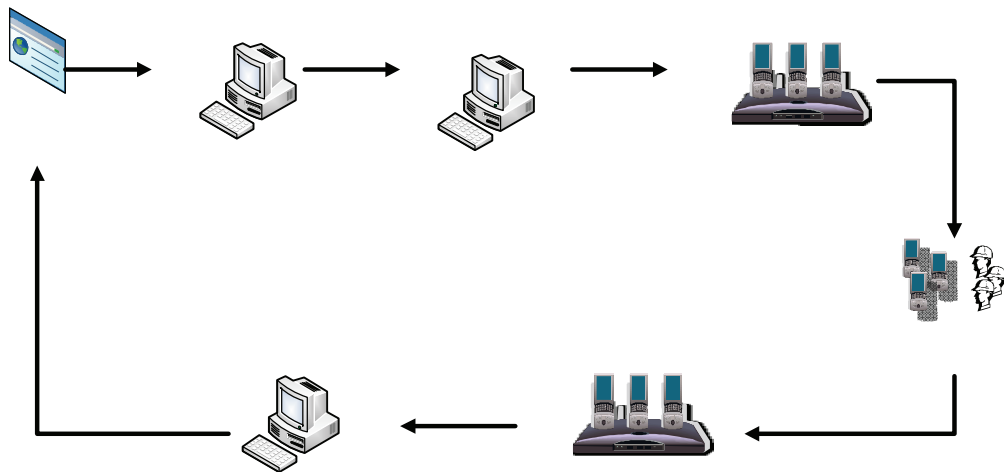
En capítulos anteriores del proyecto ya se había definido el alcance general, en este capítulo, es necesario definir los alcances funcionales del sistema informático que se implementará como parte de la ejecución del proyecto. Para definir los alcances funcionales, se analizarán los procesos ya definidos internamente por ASISTECOM.

- Macro Proceso – LECTOFACTURACION: Es el conjunto de procesos operativos que ASISTECOM ofrece a sus clientes. Estos pueden ser:



Lectura de medidores de consumo IN SITU (Lecturas), entrega de facturas de consumo (Reparto), lectura de consumo y facturación IN SITU (lecto-facturación), entre otros.

- Proceso – Lectura de medidores de consumo IN SITU: Proceso mediante el cual se recolecta información correspondiente al consumo registrado en los medidores de los abonados al servicio brindado por los clientes de ASISTECOM, principalmente empresas de servicios públicos (Empresa eléctrica, Empresa de Agua Potable).



**Figura 20. Proceso de Lectura de medidores de consumo IN SITU**

Fuente: Editado de Manuales de procedimientos ASISTECOM

Los subprocesos de la Figura 20. Proceso de Lectura de medidores de consumo IN SITU serán descompuestos en un diagrama que permita analizarlos detalladamente. Los procesos analizados son los siguientes:

- **Descarga/Cargar de Plan:** subproceso mediante el cual se descarga un archivo con los datos proporcionados por el cliente, con la lista de suministros y datos necesarios para realizar las lecturas y se carga en el sistema para su posterior asignación.
- **Asignación Rutas:** subproceso mediante el cual, las lecturas son divididas y asignadas a los Operadores lectorsitas para realizar las lecturas In Situ.
- **Sincronización de Lecturas:**
  - Pendientes: subproceso mediante el cual, las lecturas asignadas a un operador lectorsita, son sincronizadas a los dispositivos móviles de su responsabilidad para ser gestionadas en campo.
  - Gestionadas: subproceso mediante el cual, se actualizan los datos de las lecturas gestionadas.
- **Registro de Lectura:** subproceso en el cual los operadores lectorsitas toman el valor del consumo de las lecturas asignadas para su gestión y la registran en los dispositivos móviles, para su posterior sincronización.
- **Control y verificación:** subproceso mediante al cual un operador del sistema verifica la consistencia de la información recolectada por los operadores lectorsitas previo su retorno al cliente.

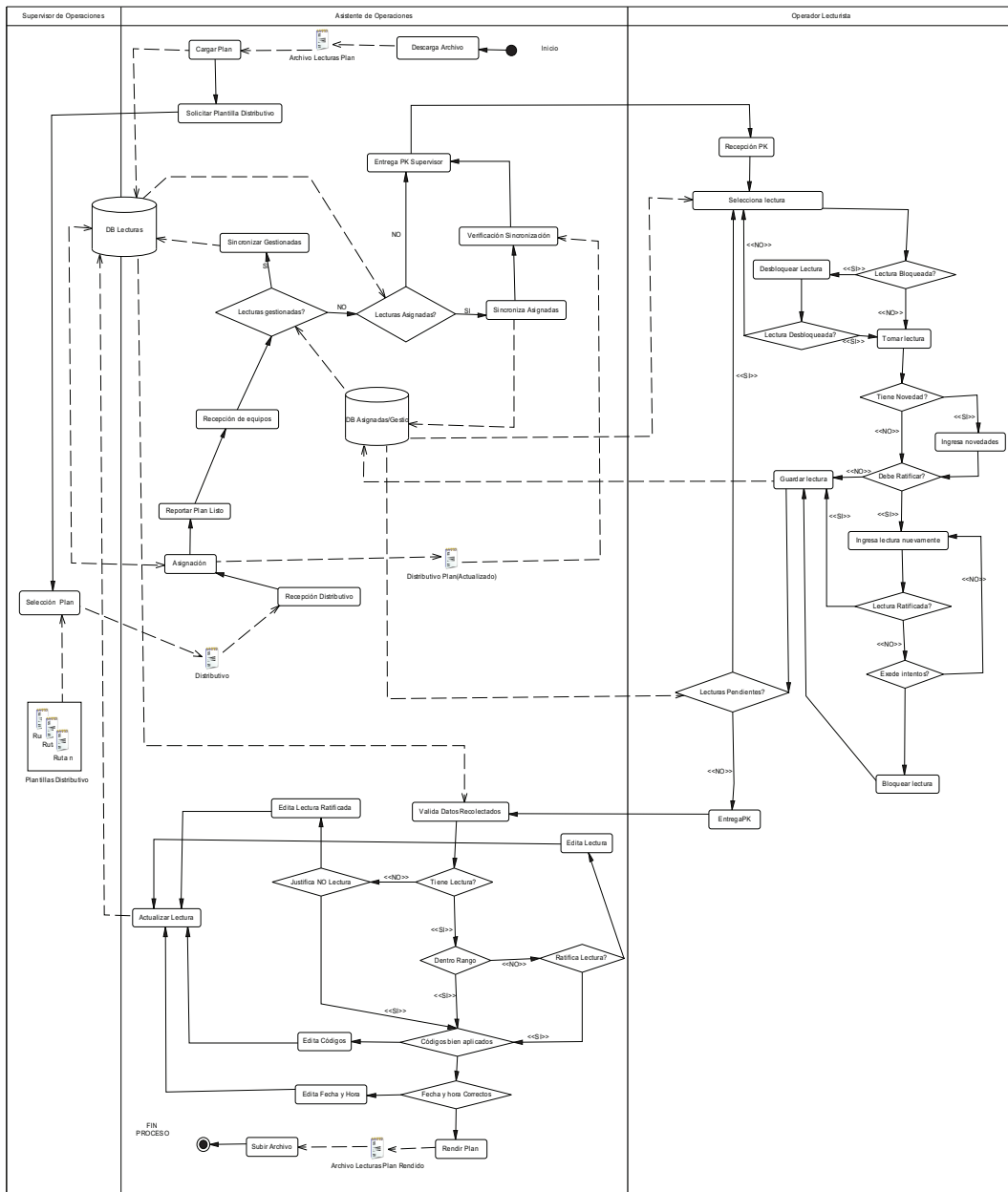

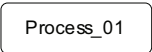
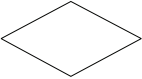

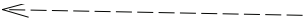

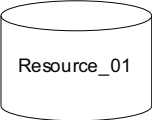


Figura 21 - Diagrama de procesos

**Tabla 6. Simbología utilizada en los diagramas**

Nombre	Símbolo	Función
Terminal		Representa el inicio y fin de un programa. También puede representar una parada o interrupción programada que sea necesaria realizar en un programa.
Proceso		Cualquier tipo de operación que pueda originar cambio de valor, formato o posición de la información
Decisión		Indica operaciones lógicas o de comparación entre datos
Indicador de dirección o línea de flujo		Indica el sentido de la ejecución de las operaciones
Indicador de dependencia		Indica que un proceso depende de un recurso (Salida)
Salida		Recurso utilizado para la ejecución de un proceso o generado durante la ejecución del proceso.
Datos Almacenados		Representa un conjunto de datos, generalmente en Base de datos.

De los procesos descritos y el respectivo diagrama, de han identificado los requisitos funcionales y no funcionales con los cuales debe cumplir el sistema a implementar y se encuentran descritos en la Tabla 7.

**Tabla 7. Requerimientos Funcionales/No Funcionales – Sprint 0**

<b>Módulo<sup>a</sup></b>	<b>Requisitos funcionales</b>	<b>Requisitos no funcionales</b>
1.1		El sistema brindará seguridad para ser utilizado por usuarios registrados del sistema, con diferentes niveles de acceso. (Este tema no fue levantado por el dueño del producto, pero es necesario)
1.2	El sistema permitirá registrar a los usuarios del sistema.	El sistema funcionará en equipos con S.O. Windows con .Net Framework 3,5
1.3	El sistema permitirá registrar los equipos utilizados para el procesos de recolección de datos en campo	
1.4	El sistema permitirá asignar al usuario responsable de un equipo.	
2.1	El sistema permitirá cargar los datos de un plan a la base de datos desde un archivo de texto (Archivo plano separado por comas [columna1],[columna2]).	Este proceso debe ser eficiente, debido a que actualmente toma demasiado tiempo. Es necesario implementar un método rápido para cargar los datos en la DB. "Se probará el método por build copy"
2.2	El sistema permitirá consultar los datos del plan cargado.	
2.3	El sistema permitirá consultar, seleccionar y asignar y reasignar rutas a gestionar (bloque de lecturas )a un usuario del sistema.	
2.4	El sistema permitirá sincronizar las lecturas asignadas a un usuario, al equipo (pocket) para su respectiva gestión en campo.	
2.5	El sistema permitirá sincronizar las lecturas gestionadas por un usuario para actualizar los datos requeridos en la base de datos.	
3.1	El sistema permitirá recolectar la información en campo, utilizando una aplicación instalada en los pockets utilizados actualmente.	La aplicación móvil funcionará el dispositivos móviles con S.O Windows Mobile 6.0 o superior con .Net Compact Framework 3,5

<b>Módulo<sup>a</sup></b>	<b>Requisitos funcionales</b>	<b>Requisitos no funcionales</b>
2.6	El sistema permitirá sincronizar los datos de las lecturas gestionadas y por gestionar utilizando la interfaz USB.	
4.1	El sistema permitirá sincronizar los datos de las lecturas gestionadas y por gestionar utilizando la red de datos proporcionada por las operadoras de telefonía celular.	La sincronización de datos se realizará utilizando un plan de datos de internet de los disponibles por las operadoras de telefonía celular, el proveedor no debe ser un limitante.
2.7	El sistema permitirá identificar quien gestiona las lecturas, en cualquiera de las formas de sincronización.	
2.8	El sistema permitirá modificar las lecturas gestionadas en campo desde la aplicación de escritorio.	
2.9	El sistema almacenara los datos de los planes gestionados permanentemente.	
5	El sistema permitirá obtener los siguientes reportes.	
5.1	*Reportes de asignación de lecturas (Distributivo Plan Actualizado con número de lecturas por lectorista)	
5.2	*Reportes de asignación de lecturas (histórico del anterior)	
5.3	*Reportes de lecturas realizadas por cada lectorista. (Detalle del Distributivo Plan Actualizado con número de lecturas por lectorista)	
5.4	*Reporte "Plan Rendido"	
5.5	Estado de sincronización (Resumen del plan rendido)	

**Nota:** La lista de requerimientos ha sido socializada con los dueños del producto, los especialistas del área de Operaciones.

<sup>a</sup> La columna identifica tentativamente el módulo en el cual se incluirán las funcionalidad. Los módulos serán a su vez tentativamente el alcance propuesto para los Sprints. Los módulos identificados durante el análisis son los siguientes:

- 1.-M. Administración
- 2.-M. Gestión (Procesamiento de la información)
- 3.-M. Gestión en campo (Aplicación para dispositivos móviles)
- 4.-M. Sincronización en línea.
- 5.-M. Reportes

#### 4.1.2. Alcance del software

De acuerdo al análisis de los procesos y las funcionalidades identificadas, el software a desarrollarse tendrá los siguientes alcances.

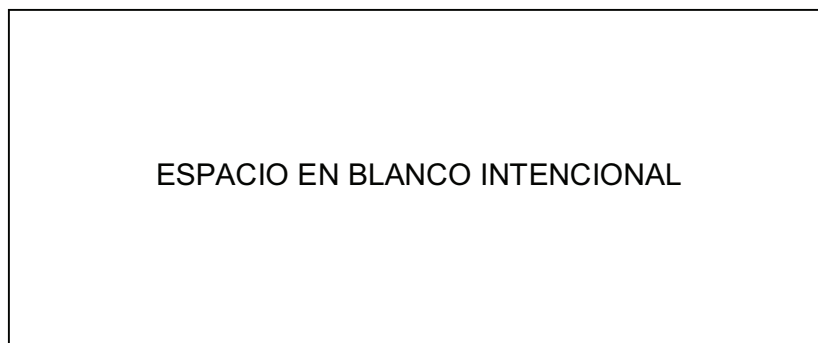
- **Módulo de administración (1 en Tabla 7)**, que permitirá:
  - Crear, eliminar, modificar usuarios, equipos y perfiles para controlar el acceso a los diferentes módulos del sistema
  
- **Módulo de Gestión (2 en Tabla 7)**, que permitirá:
  - Cargar y procesar la información proporcionada y requerida por el cliente de ASISTECOM.
  - Sincronizar desde y hacia los dispositivos móviles los datos de las lecturas gestionadas y las asignadas para su gestión, utilizando la interfaz USB de los dispositivos móviles.
  - Consultar reportes de asignación (5.1, 5.2 y 5.3 en Tabla 7. Requerimientos Funcionales/No Funcionales), los cuales permitan dar seguimiento del proceso.
  - Reporte de planes Rendidos (5.4 en Tabla 7. Requerimientos Funcionales/No Funcionales).
  
- **Módulo de Gestión en campo (3 en Tabla 7)**, aplicación para dispositivos móviles, basados en Windows Mobile que permitirá:
  - Llevar a cabo el proceso de recolección y validación de información con dispositivos móviles.

- **Módulo de Sincronización en línea (4 en Tabla 7)**, que permitirá:
  - Utilizar la red de datos proporcionada por una operadora de telefonía celular local para:
    - Descargar los datos de las lecturas asignadas a un Lectorista, para su respectiva gestión.
    - Actualizar los datos de las lecturas gestionas por el Lectorista, en el servidor central de ASISTECOM.
  - Consultar reportes de estado de sincronización (5.5 en Tabla 7. Requerimientos Funcionales/No Funcionales), los cuales permitan dar seguimiento del proceso.

#### **4.1.3. Conformación del equipo de trabajo**

Al igual que en cualquier otro tipo de proyecto, es necesario conocer el equipo humano con que se cuenta para trabajar en el proyecto.

El equipo de trabajo para llevar a cabo el Software para RECOLECCIÓN MASIVA DE INFORMACIÓN CON TECNOLOGÍA MÓVIL estará conformado según lo descrito en la tabla 8.





**Tabla 8. Equipo de Trabajo y Roles**

<b>ROL</b>	<b>Persona</b>	<b>Área <sup>a</sup></b>
<b>Product Owner</b>	Evelyn Herrera	Gerencia I.T.- ASISTECOM
<b>SCRUM Master</b>	Kleber Toapanta	Desarrollo – Outsourcing
<b>Team</b>	Kleber Toapanta	Desarrollo – Outsourcing
	Evelyn Herrera	Gerencia I.T.- ASISTECOM
	Raúl Izquierdo	Gerencia Comunicaciones - ASISTECOM
	Rubén Ortega	Soporte Técnico – Operativo - ASISTECOM
	Francisco Meza	Jefe Regional Operativo - ASISTECOM

**Nota:** Kléber Toapanta participa en más de un rol, esto es completamente factible como parte de la metodología. Esta tabla es una reproducción parcial del acta “AS-AR-PRIEE-001-2011”, en la cual se estableció el equipo de trabajo. Durante la etapa de planificación (Sprint0). Se conformará el equipo así y se definirán los roles de cada miembro en la planificación de cada Sprint del proyecto.

<sup>a</sup> Columna que identifica la área de actual de trabajo del participante del proyecto.

#### **4.1.4. Definición del Backlog del Producto.**

El Backlog Del Producto contiene toda la funcionalidad que el producto final debería tener. Tal como lo dice la metodología, para el presente proyecto se ha elaborado el Backlog del producto, identificando las funcionalidades, priorizando cada una de ellas y realizando una estimación del tiempo requerido para su implementación.

De acuerdo al estudio de los procesos y los requerimientos identificados durante la etapa de análisis, el Backlog del Producto para el presente proyecto se encuentra definido en la siguiente tabla:

**Tabla 9. Backlog Producto**

<b>ID</b>	<b>Nombre</b>	<b>Importancia<sup>a</sup></b>	<b>Tiempo estimado (semanas)<sup>b</sup></b>	<b>Comentarios</b>
1	Módulo de Administración	700	3	Funcionalidad para administración de recursos.
2	Módulo de Gestión	800	3	Funcionalidad para gestionar información (in-out) de los clientes de ASISTECOM.
3	Módulo de Gestión en Campo	900	3	Aplicación para dispositivos móviles, con funcionalidad para recolección y almacenamiento de información en campo con dispositivos móviles.
4	Sincronización en línea	1000	3	<b>OBJETIVO PRINCIPAL,</b> agilizar el proceso de sincronización de datos.

**Nota:** Esta tabla una reproducción parcial del acta “AS-AR-PRIEE-002-2011”, en la cual se estableció lista priorizada de requisitos para el presente proyecto.

<sup>a</sup> La importancia está estimada de acuerdo a las necesidades del Product Owner y esta cuantificada con números enteros entre 0 y 1000, de acuerdo a la escala de la Figura 22.

<sup>b</sup> El tiempo requerido para el desarrollo está dado principalmente por dos factores: El tiempo recomendado por SCRUM para la ejecución de cada iteración y la experiencia en la implementación de componentes de software del equipo de trabajo. En este caso cada módulo será implementado en una iteración independiente.



**Figura 22. Escala Importancia Definida por Product Owner.**

#### 4.1.5. Diseño

##### 4.1.5.1. Modelo de datos

En esta etapa se definirá en forma general, el modelo de datos que serán implementados en los siguientes Sprints. Este modelo será la guía de referencia para la implementación detallada de cada módulo.

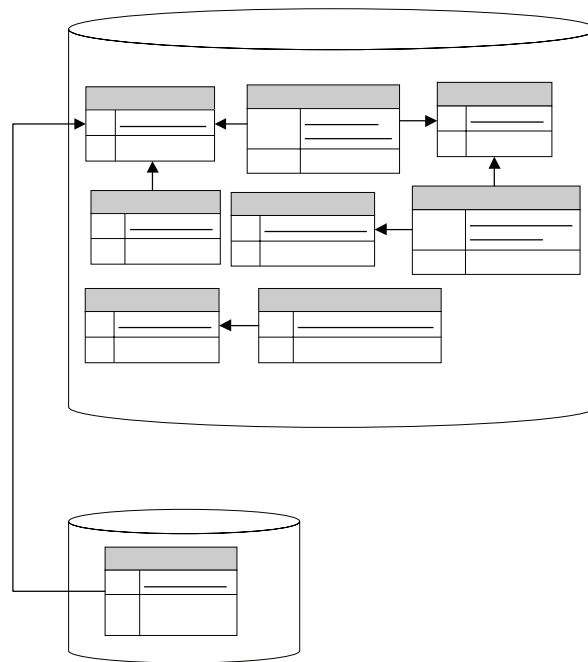


Figura 23. Modelo de Datos del Sistema

##### 4.1.5.2. Arquitectura General

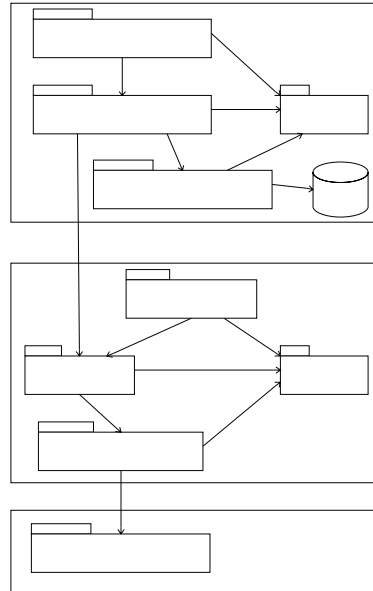
Para la implementación del sistema para RECOLECCIÓN MASIVA DE INFORMACIÓN CON TECNOLOGÍA MÓVIL, se utilizará un modelo N Capas, distribuidas de la siguiente manera:

- Capa de Presentación (GUI): presenta el sistema al usuario, muestra la información existente y captura la información del usuario con un mínimo de proceso.
- Capa de negocio (BLL): donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos del sistema.

Es posible almacenar la lógica del negocio sobre cada estación cliente, u optar por ejecuta la lógica sobre un servidor de aplicaciones. En este caso, debido al reducido número de estaciones cliente que ejecutarán esta aplicación (dos o tres estaciones cliente), es posible utilizar la primera opción. Cuando el número de clientes es considerable (superior a 10 estaciones cliente), es recomendable utilizar un servidor de aplicaciones que concentre toda la lógica del negocio.

- Capa de datos (DAL): es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.
- Capa Entidades del Negocio (BEL): Es la representación de los objetos manejados en el sistema y también de las tablas de la base de datos. Permiten el transporte de los datos desde fuera hacia la base de datos y

viceversa. Maneja el principio de programación con objetos los cuales contienen atributos que representarán datos físicos.



**Figura 24. Arquitectura Aplicación**

#### **4.1.6. Sprint1 “Módulo de Administración”**

El primer sprint tiene como objetivo implementar las funcionalidades requeridas para la administración de recursos utilizados por el sistema, esto es; Crear, eliminar, modificar usuarios, equipos y perfiles para controlar el acceso a los diferentes módulos del sistema.

##### **4.1.6.1. Planificación**

Para la planificación del Sprint1, se llevó a cabo una reunión con el Product Owner, según consta en el acta “AS-AR-PRIEE-003-2011”. En esta reunión se realiza un análisis de los procesos y las funcionalidades que serán implementados.

De la Tabla 7. Requerimientos Funcionales/No Funcionales – Sprint 0, obtenida durante el Sprint0, se selecciona las funcionalidades correspondientes al módulo que será el objetivo del Sprint1.

**Tabla 10.Requerimientos Funcionales/No Funcionales- Sprint1**

<b>REQUISITOS FUNCIONALES</b>	<b>REQUISITOS NO FUNCIONALES</b>
El sistema permitirá registrar a los usuarios del sistema.	El sistema brindará seguridad para ser utilizado por usuarios registrados del sistema, con diferentes niveles de acceso.
Los usuarios no deben ser eliminados del sistema para mantener consistencia en la información histórica.	El sistema funcionará en equipos con S.O. Windows con .Net Framework 3,5
El sistema permitirá registrar los equipos utilizados para el procesos de recolección de datos en campo	
El sistema permitirá asignar equipos registrados a usuarios registrados del sistema.	
Un usuario puede tener más de un equipo asignado y un equipo solo puede estar asignado a un usuario.	
El sistema debe manejar niveles de seguridad para diferentes grupos de usuarios.	

De acuerdo a las funcionalidades identificadas para el módulo de administración, se identifican las historias de usuario para el Sprint1.

**Tabla 11. Historias de Usuario - Sprint 1**

ID	Historia de usuario	Importancia Product Owner <sup>a</sup>	Importancia Técnica <sup>b</sup>	Descripción
1	Creación de perfiles	500	1000	Es necesario manejar niveles de acceso para los usuarios, y para ellos es importante manejar usuarios agrupados en perfiles.
2	Iniciar sesión en el sistema	500	1000	Consiste en brindar seguridad al acceso de la aplicación, permitiendo que únicamente usuarios autorizados puedan tener acceso al mismo.
3	Administración de usuarios	800	800	El sistema requiere de un número variable de usuarios con diferentes funciones, por lo que es necesario administrar los usuarios y agruparlos en perfiles con diferentes niveles de acceso a la aplicación.
4	Administración de Equipos	800	800	El sistema requiere la utilización de un número variable de equipos móviles (pockets [PK]) que puedan ser utilizados en el proceso de gestión de información en campo. Por tanto la aplicación debe registrar los equipos que se utilizan
5	Asignación de equipos	1000	800	Para que se realice el trabajo de campo, los lecturistas utilizan equipos (PK); para ello cada usuario lectorista deberá tener asignado un PK de los registrados en el sistema y con el que realizará la gestión de lecturas.

**Nota:** Esta tabla una reproducción parcial del acta "AS-AR-PRIEE-002-2011", en la cual se estableció lista priorizada de requisitos para el Sprint1.

<sup>a</sup> La importancia está estimada de acuerdo a las necesidades del Product Owner y está cuantificada con números enteros entre 0 y 1000, de acuerdo a la escala de la Figura 22.

<sup>b</sup> La importancia técnica está basada en las necesidades no funcionales desde el punto de vista del usuario, pero necesarias para un funcionamiento de la aplicación desde el punto de vista técnico.

## Definición del equipo de trabajo

El equipo de trabajo para la implementación de las funcionalidades del módulo de administración esta descrito en la siguiente tabla:

ROL		Persona	Descripción/Tareas
Product Owner		Evelyn Herrera	Administración del proyecto desde la perspectiva del negocio.
SCRUM Master		Kleber Toapanta	Asegurar que el proceso SCRUM se lleve a cabo.
Team	Codificación	Kleber Toapanta	Codificación de la las funcionalidades identificadas.
	Pruebas	Rubén Ortega	Pruebas de las funcionalidades codificadas.

### 4.1.6.2. Análisis

Una vez identificadas las historias de usuario, se identifican los actores y se diagrama los casos de uso identificados para la implementación del Sprint 1.

#### Actores del sistema

La siguiente tabla describe los actores que participan en los casos de uso identificados para el módulo de administración.

Tabla 12. Actores del sistema

Actor	Descripción
Usuario Administrador	Usuario con privilegios de administrador en el sistema. Este tipo de usuario, puede gestionar todos los recursos utilizados por el sistema



## Diagramas de casos de uso del módulo de administración

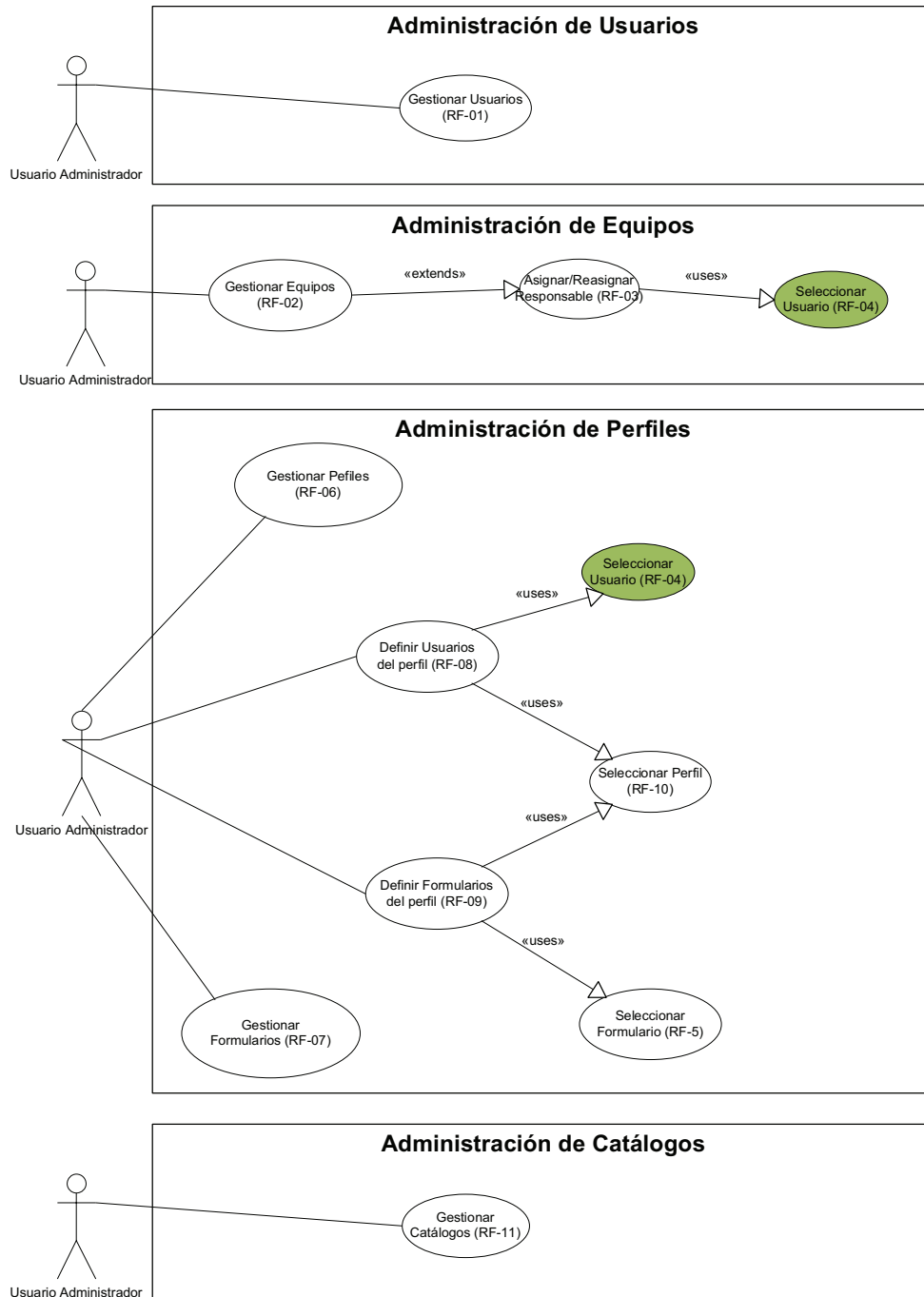


Figura 25. Casos de uso - Administración

## Especificación de casos de Uso

Los casos de uso de Figura 25 han sido especificados en las siguientes tablas:

**Tabla 13. Especificación del caso de uso: Gestionar Usuarios**

<b>ID</b>	RF-01	
<b>Descripción</b>	Proporciona funcionalidades para dar mantenimiento a los usuarios del sistema.	
<b>Precondición</b>		
<b>Postcondición</b>	Información actualizada de los usuarios del sistema.	
<b>Flujo Normal</b>	1	El usuario logeado utiliza el formulario para mantenimiento de usuarios.
	2	El sistema solicita los datos del nuevo usuario.
	3	El sistema almacena los datos proporcionados.
<b>Flujos Alternos</b>	*	<b>Usuario ya registrado</b>
		Si el usuario ya existe en el sistema y es necesario modificar la información actual, el sistema proporciona la funcionalidad para realizarlo.
<b>Excepciones</b>		
	1	Si se intenta registrar un usuario con Login, C.I. o Cód. ASISTECOM que ya se encuentren registrados, el sistema debe advertir la situación y evitará que se registre el usuario.
<b>Notas:</b>		
	1	Se requiere los siguientes datos del usuario:
		1.1. Login (*): Identificador único de cada usuario con el cual se autenticara en el sistema.
		1.2. Clave de acceso (*): código para acceso al sistema.
		1.3. Nombres (*).
	1.4. Apellidos (*).	
	1.5. Código (*): Este código es un identificador propio de la empresa y los trabajadores de ASISTECOM cuentan con uno actualmente.	
	1.6 Identificación	
	1.6.1 Tipo de identificación	
	1.7. Dirección de domicilio.	
	1.8. Dos teléfonos de contacto, además de un numero celular (*).	
	1.9. Dirección de correo electrónico.	

**Tabla 14. Especificación del caso de uso: Gestionar Equipos**

<b>ID</b>	RF-02	
<b>Descripción</b>	Proporciona funcionalidades para dar mantenimiento a los equipos (PK) utilizados para la recolección de información.	
<b>Precondición</b>		
<b>Postcondición</b>	Información actualizada de los equipos móviles disponibles para la gestión de lecturas.	
<b>Flujo Normal</b>		
	1	El usuario logeado utiliza el formulario para mantenimiento de equipos.
	2	Se ingresan los datos del nuevo equipo.
	3	El sistema almacena los datos proporcionados.
<b>Flujos Alternos</b>		
	*	<b>El equipo ya está registrado</b>
		Si el equipo ya existe en el sistema y es necesario modificar la información actual, el sistema proporciona la funcionalidad para realizarlo.
<b>Excepciones</b>		
	1	Si se intenta registrar un equipo con Número de serie o Cód. ASISTECOM que ya se encuentren registrados, el sistema debe advertir la situación y evitar que se registre el PK.
<b>Notas:</b>		
	1	Se requiere los siguientes datos:
		1.1. Código (*): Este código es un identificador único propio con los que los equipos utilizados por ASISTECOM son actualmente identificados. Este código debe ser registrado en los equipos.
		1.2. Marca (*): Se seleccionará de un catalogo de marcas.
		1.3. Modelo
		1.4. Número celular
		1.5. IMEI: Se debe revisar si es posible obtener, en la Aplicación DEMO, no fue posible de utilizar en todos los modelos probados.
		1.6. Imagen: Debe existir la posibilidad de almacenar un registro gráfico del equipo.
		1.7 Número de serie.
		1.8 Fecha de registro.
		1.9 Chip servicio internet
		Los campos macados con (*) son obligatorios.

**Tabla 15. Especificación del caso de uso: Seleccionar Usuario**

<b>ID</b>	RF-03	
<b>Descripción</b>	La asignación de equipos y lecturas requieren de la selección de un usuario registrado en el sistema, por tanto se debe poder seleccionar un usuario para utilizar en los procesos mencionados.	
<b>Precondición</b>		
	1	Usuarios registrados en el sistema.
<b>Postcondición</b>		
	?	Lecturas asignadas al usuario seleccionado.
	?	Equipo asignado al usuario seleccionado.
<b>Flujo Normal</b>		
	1	Consulta de todos los usuarios registrados en el sistema y permite filtrar de acuerdo a los perfiles existentes.
	2	Filtrar información.
	3	Seleccionar el usuario para utilizar en la tarea requerida.

**Tabla 16. Especificación del caso de uso: Asignar/Reasignar Responsable**

<b>ID</b>	RF-04	
<b>Descripción</b>	Proporciona funcionalidades para asignar al usuario responsable de un equipo (PK).	
<b>Precondición</b>		
	1	Usuarios registrados en el sistema.
	2	Equipos registrados en el sistema en estado activo.
<b>Postcondición</b>	Información actualizada de los usuarios responsables de los equipos (PK).	
<b>Flujo Normal</b>		
	1	El usuario logeado selecciona un equipo de los registrados en el sistema.
	2	El equipo no tiene un usuario asignado.
	3	Se selecciona un usuario del sistema.
	4	El sistema registra la asignación del equipo al usuario.
<b>Flujos Alternos</b>		
	*	<b>Reasignación</b>
	1	El usuario logeado selecciona un equipo de los registrados en el sistema y este tiene asignado un usuario.
	2	Se selecciona otro usuario del sistema.
	3	El sistema actualiza la asignación del equipo al usuario.
	*	<b>Liberación</b>
	1	El usuario logeado selecciona un equipo de los registrados en el sistema y este tiene asignado un usuario.
	2	Se elimina la asignación, dejando el equipo libre para asignar a otro usuario.
	3	Se guardará un log de los eventos (asignación/reasignación).
<b>Notas:</b>		
	1. Un equipo debe ser asignado únicamente a un usuario.	
	2. Un usuario puede tener asignado más de un equipo.	
	3. Debe ser posible exportar a Excel una lista con los detalles de la asignación.	
	4. El proceso de asignación debe ser lo más fácil posible, debido a la alta frecuencia con que los usuarios cambian de equipos.	

**Tabla 17. Especificación del caso de uso: Seleccionar Perfil**

<b>ID</b>	RF-05	
<b>Descripción</b>	Consiste en consultar y seleccionar un perfil determinado, para asignar usuarios o asignar formularios.	
<b>Precondición</b>		
	1	Perfiles registrados en el sistema.
<b>Postcondición</b>		
	1	Perfil seleccionado para ser utilizado en diferentes procesos.
<b>Flujo Normal</b>		
	1	Consulta de todos los perfiles registrados en el sistema.
	2	Filtrar información.
	3	Seleccionar el perfil para utilizar en la tarea requerida.

**Tabla 18. Especificación del caso de uso: Seleccionar Formulario**

<b>ID</b>	RF-06	
<b>Descripción</b>	Consiste en consultar y seleccionar un formulario (s), para asignar a un determinado perfil.	
<b>Precondición</b>		
	1	Formularios registrados en el sistema.
<b>Postcondición</b>		
	?	Formulario asignado al perfil.
<b>Flujo Normal</b>		
	1	Consulta de todos los formularios registrados en el sistema.
	2	Filtrar información.
	3	Seleccionar formularios a asignar a un perfil.

**Tabla 19. Especificación del caso de uso: Gestionar Perfiles**

<b>ID</b>	RF-07	
<b>Descripción</b>	Proporciona funcionalidades crear, modificar perfiles de usuario, los cuales proporcionarán diferentes niveles de acceso a la aplicación.	
<b>Precondición</b>		
<b>Postcondición</b>	Información actualizada de los perfiles disponibles en el sistema.	
<b>Flujo Normal</b>	1	El usuario logeado ingresa los datos de un perfil.
	2	El sistema guarda la información de nuevo perfil.
<b>Flujos Alternos</b>	*	<b>El perfil ya existe</b>
	1	El usuario logeado selecciona un perfil existente
	2	Se modifica la información del perfil
	3	El sistema actualiza la información del perfil modificado.
	*	<b>Eliminar perfil</b>
	1	El usuario logeado selecciona un perfil existente
	2	Se solicita la eliminación del perfil.
	3	El sistema elimina de la DB los datos del perfil seleccionado.

**Tabla 20. Especificación del caso de uso: Gestionar Formularios**

<b>ID</b>	RF-08	
<b>Descripción</b>	Proporciona funcionalidades para registrar información de los formularios (pantallas) existentes en el sistema. Estas son posteriormente asignadas a los perfiles existentes, para brindar acceso a las pantallas requeridas en un perfil.	
<b>Precondición</b>		
	1	Desarrollo de los componentes GUI, con formularios utilizados en el sistema.
<b>Postcondición</b>	Información actualizada con los datos de los formularios (pantallas) disponibles en el sistema.	
<b>Flujo Normal</b>		
	1	El usuario selecciona un componente GUI (Ejem: RMI.GUI*.DLL)
	2	El sistema muestra todos los formularios disponibles en el componente.
	3	El usuario selecciona un formulario.
	4	El usuario ingresa una descripción que permita identificar la utilidad que tiene el formulario dentro del sistema, el módulo en el cual es utilizado y el estado (Activo/Inactivo).
	5	El sistema registra el nuevo formulario disponible en el sistema.
<b>Flujos Alternos</b>		
	*	<b>El formulario ya existe</b>
	1	El usuario logeado selecciona un formulario existente.
	2	Se modifica la información del formulario.
	3	El sistema actualiza la información del formulario modificado.
	*	<b>Eliminar Formulario</b>
	1	El usuario logeado selecciona un formulario existente.
	2	Se solicita la eliminación del formulario.
	3	El sistema elimina de la DB los datos del formulario seleccionado.



**Tabla 21. Especificación del caso de uso: Definir Usuarios del perfil**

<b>ID</b>	RF-09	
<b>Descripción</b>	Proporciona funcionalidades para asignar usuarios a un determinado perfil.	
<b>Precondición</b>		
	1	Usuarios registrados en el sistema.
	2	Perfiles registrados en el sistema.
<b>Postcondición</b>	Usuarios asignados a un perfil de seguridad.	
<b>Flujo Normal</b>		
	1	El usuario logeado selecciona perfil.
	2	El usuario selecciona los usuarios para asignarlos al perfil.
	3	El sistema almacena la información de los usuarios asignados al perfil.
<b>Flujos Alternos</b>		
	*	<b>Los usuarios ya han sido asignados, se requiere removerlos.</b>
	1	El usuario logeado selecciona perfil.
	2	El usuario selecciona los usuarios que no deben pertenecer al perfil
	3	El sistema almacena la información de los usuarios asignados al perfil.

**Tabla 22. Especificación del caso de uso: Definir Formularios del perfil**

<b>ID</b>	RF-10	
<b>Descripción</b>	Proporciona funcionalidades para asignar formularios a un determinado perfil.	
<b>Precondición</b>		
	1	Usuarios registrados en el sistema.
	2	Formularios registrados en el sistema.
<b>Postcondición</b>	Formularios asignados a un perfil de seguridad.	
<b>Flujo Normal</b>		
	1	El usuario logeado selecciona perfil.
	2	El usuario consulta los formularios existentes.
	3	El usuario selecciona los formularios.
	4	El sistema guarda la información de los formularios asignados al perfil.
<b>Flujos Alternos</b>		
	*	<b>Los formularios ya han sido asignados, se requiere removerlos.</b>
	1	El usuario logeado selecciona perfil.
	2	El usuario consulta los formularios asignados al perfil.
	3	El usuario selecciona los formularios que deben ser removidos del perfil
	4	El sistema guarda la información de los formularios asignados al perfil.

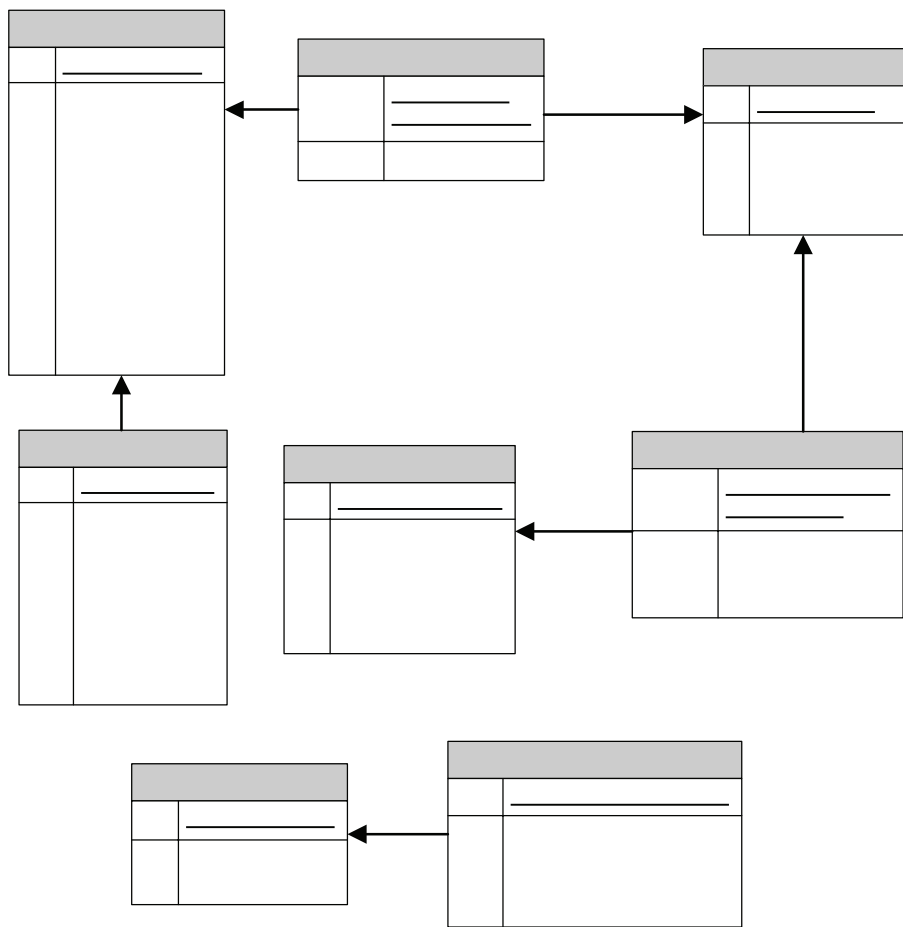
**Tabla 23. Especificación del caso de uso: Gestión de Catálogos**

<b>ID</b>	RF-11								
<b>Descripción</b>	Proporciona funcionalidades para crear, modificar catálogos utilizados en el sistema. Esto permite agregar o eliminar dinámicamente listas de valores utilizados como catálogos.								
<b>Precondición</b>									
<b>Postcondición</b>	<table border="1"> <tr> <td>1</td> <td>Catálogos Actualizados</td> </tr> <tr> <td>2</td> <td>Detalles de catálogos actualizados.</td> </tr> </table>	1	Catálogos Actualizados	2	Detalles de catálogos actualizados.				
1	Catálogos Actualizados								
2	Detalles de catálogos actualizados.								
<b>Flujo Normal</b>	<table border="1"> <tr> <td>1</td> <td>Ingresar los datos del catálogo (Descripción de las listas de valores).</td> </tr> <tr> <td>2</td> <td>El sistema guarda los datos del catálogo.</td> </tr> <tr> <td>3</td> <td>Ingresar detalles del catálogo (Los detalles corresponden a la lista de valores del catálogo).</td> </tr> <tr> <td>4</td> <td>El sistema almacena la lista de detalles del catálogo.</td> </tr> </table>	1	Ingresar los datos del catálogo (Descripción de las listas de valores).	2	El sistema guarda los datos del catálogo.	3	Ingresar detalles del catálogo (Los detalles corresponden a la lista de valores del catálogo).	4	El sistema almacena la lista de detalles del catálogo.
1	Ingresar los datos del catálogo (Descripción de las listas de valores).								
2	El sistema guarda los datos del catálogo.								
3	Ingresar detalles del catálogo (Los detalles corresponden a la lista de valores del catálogo).								
4	El sistema almacena la lista de detalles del catálogo.								
<b>Flujos Alternos</b>	<table border="1"> <tr> <td>*</td> <td><b>El catálogo ya existe</b></td> </tr> <tr> <td>1</td> <td>El usuario logeado selecciona un catálogo existente.</td> </tr> <tr> <td>2</td> <td>Se modifica la información del catálogo y el sistema almacena la información modificada (opcional)</td> </tr> <tr> <td>3</td> <td>Se actualiza la información de los detalles del catálogo. La actualización puede incluir la adición, eliminación o actualización de los detalles (Opcional).</td> </tr> </table>	*	<b>El catálogo ya existe</b>	1	El usuario logeado selecciona un catálogo existente.	2	Se modifica la información del catálogo y el sistema almacena la información modificada (opcional)	3	Se actualiza la información de los detalles del catálogo. La actualización puede incluir la adición, eliminación o actualización de los detalles (Opcional).
*	<b>El catálogo ya existe</b>								
1	El usuario logeado selecciona un catálogo existente.								
2	Se modifica la información del catálogo y el sistema almacena la información modificada (opcional)								
3	Se actualiza la información de los detalles del catálogo. La actualización puede incluir la adición, eliminación o actualización de los detalles (Opcional).								

#### 4.1.6.3. Diseño

##### Modelo de datos

Del análisis de la especificación de los casos de uso, se determina la necesidad de utilizar el modelo de datos descrito en el diagrama de entidad relación de la Figura 26. Diagrama Entidad Relación-Módulo Administración, las mismas que serán implementadas como objetos en la base de datos y como entidades del negocio en la codificación de la aplicación.



**Figura 26. Diagrama Entidad Relación-Módulo Administración**

#### **4.1.6.4. Arquitectura**

De la arquitectura planteada inicialmente para la aplicación completa (Figura 24. Arquitectura Aplicación), la codificación para el módulo de administración se implementará de la siguiente manera:

GUI: Formularios (pantallas) para:

- Formularios para administración de recursos
  - Administración de Usuarios
  - Administración de Equipos
  - Administración de perfiles

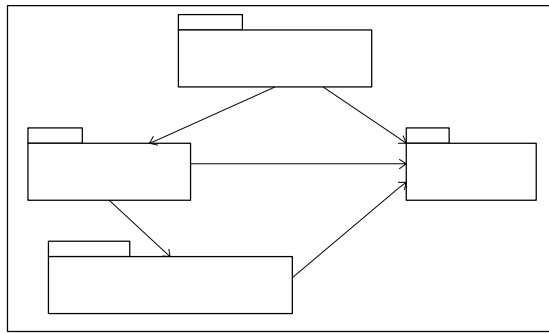
- Asignación de usuarios al perfil
- Asignación de formularios al perfil
- Formularios comunes
  - Principal
  - Login
  - Mensajes
- Formulario para configuración
  - Configurar conexión a DB

BLL: Componentes para controlar la lógica de:

- Administración de recursos
  - Administración de Usuarios
  - Administración de Equipos
  - Administración de perfiles
  - Asignación de usuarios al perfil
  - Asignación de formularios al perfil
- Lógica común
  - Seguridad
  - Login

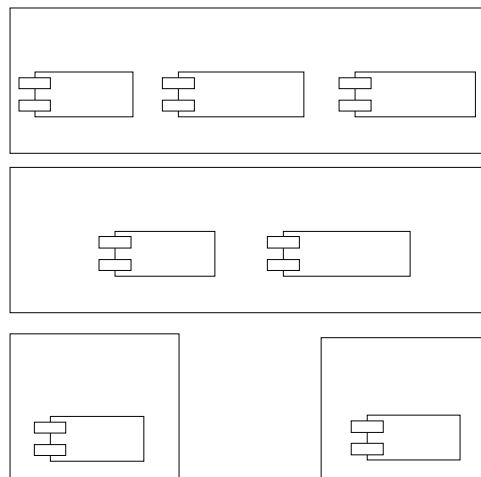
DAL: Componente para acceso a la capa de datos (DB)

BEL: Componente para administrar las entidades utilizadas en el sistema



**Figura 27. Arquitectura Aplicación de escritorio – Módulo de Administración**

Los siguientes diagramas describirán de manera detallada como se implementarán los requerimientos del módulo de administración.



**Figura 28. Componentes - Módulo Administración**

#### **4.1.6.5. Construcción y Pruebas**

##### **Backlog del Sprint**

Una vez identificadas las historias de usuario, las cuales se encuentran descritas en los casos de uso y sus respectivas especificaciones, estas serán implementadas en el primer sprint; para ello es necesario definir el Backlog del sprint (pila de tareas) que permitan implementar las funcionalidades especificadas

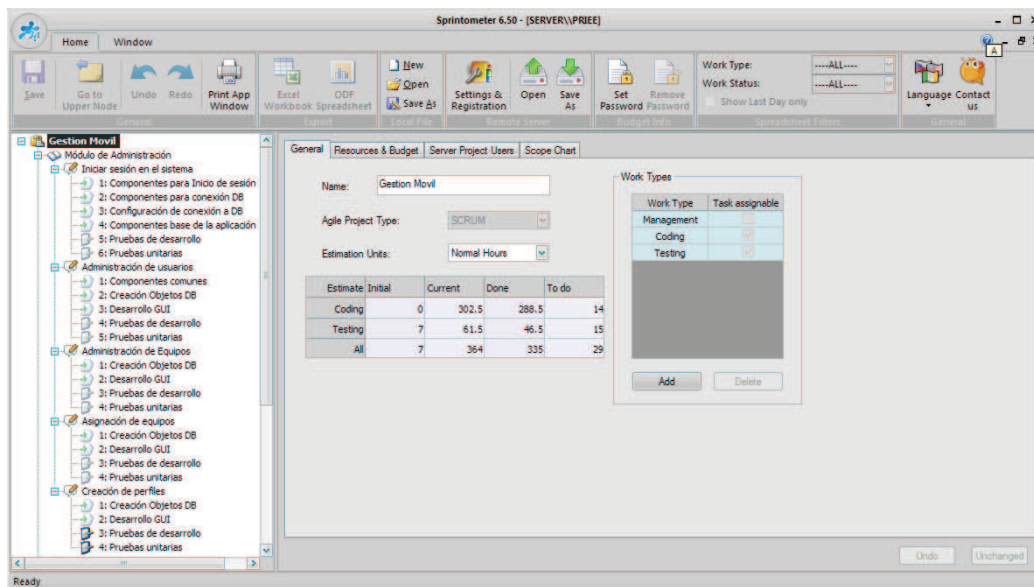
en etapa de análisis, aplicando diseño planteado. La pila de tareas para el Sprint1 se muestra en la siguiente tabla:

**Tabla 24. Backlog Sprint “Módulo de Administración”**

Story ID	Task#	Story Name, Task Name	Assigned 1	Estimado (Horas)
<b>1</b>		<b>Iniciar sesión en el sistema</b>		
	1	Componentes para Inicio de sesión	Kleber Toapanta	8
	2	Componentes para conexión DB	Kleber Toapanta	8
	3	Configuración de conexión a DB	Kleber Toapanta	6
	4	Componentes base de la aplicación	Kleber Toapanta	8
	5	Pruebas de desarrollo	Kleber Toapanta	2
	6	Pruebas unitarias	Rubén Ortega	1
<b>2</b>		<b>Administración de usuarios</b>		
	1	Componentes comunes	Kleber Toapanta	8
	2	Creación Objetos DB	Kleber Toapanta	5
	3	Desarrollo GUI	Kleber Toapanta	8
	4	Pruebas de desarrollo	Kleber Toapanta	2
	5	Pruebas unitarias	Rubén Ortega	1
<b>3</b>		<b>Administración de Equipos</b>		
	1	Creación Objetos DB	Kleber Toapanta	5
	2	Desarrollo GUI	Kleber Toapanta	5
	3	Pruebas de desarrollo	Kleber Toapanta	2
	4	Pruebas unitarias	Rubén Ortega	1
<b>4</b>		<b>Asignación de equipos</b>		
	1	Creación Objetos DB	Kleber Toapanta	3
	2	Desarrollo GUI	Kleber Toapanta	3
	3	Pruebas de desarrollo	Kleber Toapanta	1
	4	Pruebas unitarias	Rubén Ortega	1
<b>5</b>		<b>Creación de perfiles</b>		
	1	Creación Objetos DB	Kleber Toapanta	4
	2	Desarrollo GUI	Kleber Toapanta	5
	3	Pruebas de desarrollo	Kleber Toapanta	2
	4	Pruebas unitarias	Rubén Ortega	1

**Nota:** Esta tabla una reproducción parcial del reporte exportado desde la herramienta “Sprint to meter” utilizada para gestionar el presente proyecto.

Para llevar un control integral de los avances de las tareas planteadas para implementar las historias de usuario definida en la planificación del sprint, las tareas son subidas a la herramienta definida. En este caso “Sprint to meter”.



**Figura 29. Backlog Sprint 1- Sprint to Meter**

Fuente: Captura pantalla herramienta “Sprint to Meter”

## Pruebas

Como se había expuesto en el CAPÍTULO 1, sección de Construcción y Pruebas, las pruebas a realizar dependerán del módulo y componentes de los mismos. Los resultados de las pruebas serán adjuntados como anexos al presente documento.

En ese contexto, los métodos de pruebas utilizados en este módulo son:

GUI: Formularios (pantallas) para:

- Formularios para administración de recursos -> Pruebas de Aceptación



- Formularios comunes -> Pruebas de Aceptación
- Formulario para configuración -> Pruebas de Aceptación

BLL: Componentes para controlar la lógica de:

- Administración de recursos -> Pruebas de unidad (Caja Blanca)
- Lógica común -> Pruebas de unidad (Caja Blanca)

DAL: Componente para acceso a la capa de datos (DB) -> Pruebas de unidad (Caja Blanca, Caminos)

BEL: Componente para administrar las entidades. -> Pruebas de unidad

Para agilizar la etapa de pruebas, la persona asignada del equipo, utilizará la herramienta JIRA, con esta gestionara y dará seguimiento a los errores presentados durante la etapa de codificación.



**Figura 30. Reporte de errores – JIRA**

Fuente: Captura pantalla herramienta “JIRA”

#### 4.1.7. Sprint2 “Módulo de Gestión”

El segundo sprint tiene como objetivo implementar las funcionalidades requeridas para la gestión planes de lectura, esto es; Cargar planes de trabajo en el sistema, asignar/ reasignar lecturas a usuarios para su posterior gestión, sincronización de datos (gestionados y pendientes) y actualización de lecturas con los datos recolectados durante la gestión IN SITU.

##### 4.1.7.1. Planificación

De la Tabla 7. Requerimientos Funcionales/No Funcionales – Sprint 0, obtenida durante el Sprint0, se selecciona las funcionalidades correspondientes al módulo de gestión que serán implementadas en este Sprint.

**Tabla 25.Requerimientos Funcionales/No Funcionales- Sprint 2**

REQUISITOS FUNCIONALES	REQUISITOS NO FUNCIONALES
El sistema permitirá cargar los datos de un plan a la base de datos desde un archivo de texto (Archivo plano separado por comas [columna1],[columna2]).	Este proceso debe ser eficiente, debido a que actualmente toma demasiado tiempo. Es necesario implementar un método rápido para cargar los datos en la DB. “Se probara el método por build copy”
El sistema permitirá consultar los datos del plan cargado.	
El sistema permitirá consultar, seleccionar y asignar y reasignar rutas a gestionar (bloque de lecturas )a un usuario del sistema.	
El sistema permitirá sincronizar las lecturas asignadas a un usuario, al equipo (pocket) para su respectiva gestión en campo.	
El sistema permitirá sincronizar las lecturas gestionadas por un usuario.	

Una vez identificados los requerimientos funcionales y no funcionales se plantean las historias de usuario que deben ser implementadas durante el desarrollo del Sprint2.

**Tabla 26.Historias de Usuarios – Sprint 2**

<b>ID</b>	<b>Historia de usuario</b>	<b>Importancia Product Owner<sup>a</sup></b>	<b>Importancia Técnica<sup>b</sup></b>	<b>Descripción</b>
1	Carga Plan Trabajo	900	1000	Permite cargar los datos de las lecturas proporcionadas por el cliente. Se buscara la mejor alternativa para optimizar el proceso. (La opción propuesta es usar SQLXML)
2	Asignación Rutas- Usuarios	800	800	Consiste en asignar lecturas de un plan de trabajo a los usuarios lectors, para su posterior gestión en campo.
3	Sincronización	800	800	Consiste en actualizar la información recolectada en campo con los dispositivos móviles y en enviar a los pocket la información de las lecturas pendientes de gestión en campo.

**Nota:**

<sup>a</sup> La importancia está estimada de acuerdo a las necesidades del Product Owner y está cuantificada con números enteros entre 0 y 1000, de acuerdo a la escala de la Figura 22.

<sup>b</sup> La importancia técnica está basada en las necesidades no funcionales desde el punto de vista del usuario, pero necesarias para un funcionamiento de la aplicación desde el punto de vista técnico.