SRS: SYSTEM REQUERIMENTS SPECIFICATION

**2010**

Alimnova

01/01/2010

****

**LAURA ARIAS PRADA**

**NESTOR DIAZGRANADOS**

**ANDREA FAJARDO**

**WILLIAM JIMÉNEZ**

**GERMÁN MORALES**

**DAVID SUAREZ**

**HISTORIAL DE CAMBIOS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Versión | Fecha | Sección del documento modificada | Descripción de cambios (corta) | Responsable (S) |
| 0.1.0 | **Marzo 12/2010** | **Sección 1,2,3** | **Alteración de las subsecciones a nuestro proyecto** | Todos |
| 0.2.0 | **Marzo 13/2010** | **Sección 2.5, 2.7** | **Avance de modelo de dominio y distribución de requerimientos** | Andrea Fajardo, Arquitecto; Néstor Diazgranados, Diseñador Gráfico. |
| 0.3.0 | Marzo 14/2010 | Sección 1.6 | Descripción del modelo en espiral | Laura Arias, Director de proyectos |

*Tabla 1: historial de cambios*

**CONTENIDO**

[HISTORIAL DE CAMBIOS 1](#_Toc256139151)

[Lista de Tablas 2](#_Toc256139152)

[Lista de Ilustraciones 4](#_Toc256139153)

[1. INTRODUCCIÓN 5](#_Toc256139154)

[1.1 PROPÓSITO 5](#_Toc256139155)

[1.2 ALCANCE 5](#_Toc256139156)

[1.3 DEFINICIONES, ACRÓNIMOS, Y ABREVIACIONES 6](#_Toc256139157)

[1.4 REFERENCIAS 7](#_Toc256139158)

[1.5 APRECIACIÓN GLOBAL 7](#_Toc256139159)

[1.6 MODELO DE ESPIRAL PARA EL PROCESO DE INGENIERÍA DE REQUERIMIENTOS 8](#_Toc256139160)

[2. DESCRIPCIÓN GLOBAL 9](#_Toc256139161)

[2.1 PERSPECTIVA DEL PRODUCTO 9](#_Toc256139162)

[2.1.1 INTERFACES CON EL SISTEMA 9](#_Toc256139163)

[2.1.2 INTERFACES CON EL USUARIO 10](#_Toc256139164)

[2.1.3 INTERFACES CON EL HARDWARE 12](#_Toc256139165)

[2.1.4 INTERFACES CON EL SOFTWARE 13](#_Toc256139166)

[2.1.5 INTERFACES DE COMUNICACIÓN 17](#_Toc256139167)

[2.1.6 RESTRICCIONES 17](#_Toc256139168)

[2.1.7 OPERACIONES 18](#_Toc256139169)

[2.1.8 REQUERIMIENTOS DE ADAPTACIÓN DEL SITIO 18](#_Toc256139170)

[2.2 FUNCIONES DEL PRODUCTO 18](#_Toc256139171)

[2.3 CARACTERÍSTICAS DEL USUARIO FINAL 19](#_Toc256139172)

[2.4 RESTRICCIONES 21](#_Toc256139173)

[2.5 MODELO DEL DOMINIO 22](#_Toc256139174)

[2.6 SUPOSICIONES Y DEPENDENCIAS 24](#_Toc256139175)

[2.7 DISTRIBUCIÓN DE REQUERIMIENTOS 26](#_Toc256139176)

[3. REQUERIMIENTOS ESPECÍFICOS 29](#_Toc256139177)

[3.1 REQUERIMIENTOS DE INTERFACES EXTERNAS 34](#_Toc256139178)

[3.1.1 INTERFACES CON EL USUARIO 34](#_Toc256139179)

[3.1.2 INTERFACES CON EL HARDWARE 35](#_Toc256139180)

[3.1.3 INTERFACES CON EL SOFTWARE 36](#_Toc256139181)

[3.1.4 INTERFACES DE COMUNICACIONES 37](#_Toc256139182)

[3.2 CARACTERÍSTICAS DEL PRODUCTO DE SOFTWARE 37](#_Toc256139183)

[3.3 REQUERIMIENTOS DE DESEMPEÑO 39](#_Toc256139184)

[3.4 RESTRICCIONES DE DISEÑO 41](#_Toc256139185)

[3.5 ATRIBUTOS DEL SISTEMA DE SOFTWARE (NO FUNCIONALES) 42](#_Toc256139186)

[3.5.1 CONFIABILIDAD 42](#_Toc256139187)

[3.5.2 DISPONIBILIDAD 42](#_Toc256139188)

[3.5.3 SEGURIDAD 43](#_Toc256139189)

[3.5.4 MANTENIBILIDAD 43](#_Toc256139190)

[3.5.5 PORTABILIDAD 44](#_Toc256139191)

[3.6 REQUERIMIENTOS DE LA BASE DE DATOS 45](#_Toc256139192)

[4. ANEXOS 47](#_Toc256139193)

**LISTA DE TABLAS**

[Tabla 1: Historial de cambios 1](#_Toc176967938)

[Tabla 2: Acrónimos 7](#_Toc176967939)

[Tabla 3: Interfaces con el Software 13](#_Toc176967940)

[Tabla 4: Descripción de las Características del Usuario 17](#_Toc176967941)

[Tabla 5: Definiciones del Modelo de Dominio 19](#_Toc176967942)

[Tabla 6: Formato de documentación del Modelo del Dominio 20](#_Toc176967943)

[Tabla 7: Documentación de Requerimientos 27](#_Toc176967944)

**LISTA DE ILUSTRACIONES**

Ilustración 1: Propósito 5

Ilustración 2: Alcance 6

Ilustración 3: Apreciación Global 7

Ilustración 4: Tipos de productos de software 8

Ilustración 5: Interfaces con el usuario 10

Ilustración 6: Operaciones 15

Ilustración 7: Tips para definir funciones del producto 16

Ilustración 8: Características del Usuario 17

Ilustración 9: Restricciones 18

Ilustración 10: Limitaciones 18

Ilustración 11: Descripción documentación del modelo del dominio 20

Ilustración 12: Suposiciones 21

Ilustración 13: Dependencias [1] 21

Ilustración 14: Distribución de requerimientos 23

Ilustración 15: Características de los Requerimientos 26

Ilustración 16: Documentación de Requerimientos 28

Ilustración 17: Interfaces con el Usurario 29

Ilustración 18: Interfaces de Hardware 30

Ilustración 19: Interfaces con el Software 31

Ilustración 20: División por Funcionalidades 32

Ilustración 21: Ejemplo Enunciado Requerimientos 34

Ilustración 22: Atributos de Calidad a tener en cuenta 35

Ilustración 23: Características de Confiabilidad 36

Ilustración 24: Características de Disponibilidad 37

Ilustración 25: Características de Seguridad 37

Ilustración 26: Características de Mantenibilidad 38

Ilustración 27: Portabilidad del Sistema 38

Ilustración 28: Características Bases de Datos 39

# **INTRODUCCIÓN**

## **PROPÓSITO**

*En esta sección se describe la intención con la que se realiza el documento, es decir, se deben exponer las razones por las que es importante, identificando el producto de software al cual se le van a especificar los requerimientos, la audiencia que está interesada en el contenido y el alcance del documento, es decir, especificar el alcance del producto que será descrito [1]. En la ilustración 1 se resume el contenido de esta sección.*

*Ilustración 1: Propósito*

## **ALCANCE**

*Se describe el alcance del producto, es decir, la sección contiene una breve descripción del producto de software sobre el cual se realiza el SRS, indicando su nombre, las funcionalidades que incluirá y su utilidad (objetivos, beneficios). También puede ser incluida la relación entre el producto y las metas corporativas o estrategia de negocio resaltando la importancia que tiene para la organización [1].*

Ilustración 2: Alcance

## **DEFINICIONES, ACRÓNIMOS, Y ABREVIACIONES**

*Se deben especificar los términos clave que serán utilizados en el documento con el fin de aclarar el contenido y asegurar su óptimo entendimiento. Se recomienda reutilizar los especificados en el documento SPMP y formar un glosario general de términos. Para mayor claridad revisar la sección 3 de la plantilla del SPMP [2]. En la tabla 2 se presentan algunos ejemplos de acrónimos para el documento SRS.*

|  |  |
| --- | --- |
| API | Aplication Programming Interface |
| CRUD | Create, Retrieve, Update, Delete |
| DBMS | Data Base Management System |
| GNU | Es un acrónimo recursivo que significa "GNU No es Unix" |
| [GNU](http://es.wikipedia.org/wiki/GNU) GPL | General Public License o licencia pública general |
| JDBC | Java DataBase Connectivity |
| JVM | Java Virtual Machine |
| LAN | Local Area Network |
| PHP | Es un acrónimo recursivo “Hypertext Preprocessor” |
| RFC | Request For Comments |
| SDD | Software Design Description |
| SQL | Structured Query Language |
| SRS | Software Requirement Specification |
| WLAN | Wireless Local Area Network |
| WPA | World Poker Association |

Tabla 2: Acrónimos

## **REFERENCIAS**

*Remítase a la sección 2 de la plantilla del SPMP [2] donde se encuentra explicado de manera detallada el manejo de referencias.*

## **APRECIACIÓN GLOBAL**

*Esta sección debe tener una descripción general del contenido del documento, especificando su organización o distribución. Es una guía para el lector que le permitirá estructurar la información que encontrará en del documento.*

Ilustración 3: Apreciación Global

## **MODELO DE ESPIRAL PARA EL PROCESO DE INGENIERÍA DE REQUERIMIENTOS**

Nuestro modelo general del proyecto es espiral. La grafica que se mostrara a continuación describirá las iteraciones durante el proceso de requerimientos.

Ilustración 4: Modelo espiral para el proceso de ingeniería de requerimientos. Tomado de SWEBOK: Software Requirementes Engineering Knowledge Area Description

La descripción de cada fase se hará a continuación:

**ELICITACIÓN DE REQUERIMIENTOS:**

También se denomina descubrimiento o adquisición de requerimientos. Se preocupa por definir de donde vienen los requerimientos y como pueden ser clasificados. Es el primer paso para la comprensión del problema.

Este paso se divide en 2 tópicos:

* Fuentes de los requerimientos: Objetivos por los cuales se hacen los requerimientos. Dominio del conocimiento: se debe tener conocimiento de la aplicación a realizar. Ambiente operacional: la aplicación en que software se aplicara, cuales son sus tiempos reales. Ambiente organizacional: los requerimientos no deben afectar los procesos de negocio de quien desea poseer la aplicación
* Técnicas de elicitación: Entrevistas al cliente. Escenarios que consiste en diseñar un modelo que permita generar preguntas de que pasaría si alguna cosa no se cumple. Prototipos. Observación.

**ANÁLISIS DE REQUERIMIENTOS**

Se concentra en la detección y resolución de conflictos entre los requerimientos. Descubrir los límites del sistema. Transformar los requerimientos de sistema a requerimientos de software. Se estudian las siguientes características:

**Clasificación de requerimientos:**

* ¿Es funcional o no es funcional?
* ¿Es derivado de otro(s) requerimiento(s)?
* ¿Cuál es la prioridad del requerimiento?
* El ámbito del requerimiento, es decir, la categoría
* La estabilidad del requerimiento que no afecte si se modifica o elimina.

**Modelo conceptual:**

El desarrollo del modelo conceptual es fundamental en el análisis de requerimientos, ya que es el inicio de la solución. Este modelo incluye los flujos de control, los datos, la interacción de los usuarios. La selección del modelo depende de varios factores entre ellos:

* La naturaleza del problema
* La experiencia del analista de requerimientos
* Los requerimientos del proceso del cliente
* Disponibilidad de los métodos y herramientas

**Diseño arquitectónico y asignación de requerimientos:**

Incluye la descripción de subsistemas y componentes que cumplan con los requerimientos especificados. Permite un análisis detallado de las necesidades. Los requerimientos asignados a un componente específico.

**Negociación de requerimientos**

Es la resolución de un conflicto entre el cliente y lo que se puede realmente hacer.

**ESPECIFICACIÓN DE REQUERIMIENTOS DE SOFTWARE**

Este tópico está ligado a la estructura, calidad y verificación del documento de requerimientos. Existen 2 documentos esenciales y otros recursos:

* Documentación de los requerimientos. Es decir las plantillas
* SRS
* IEEE std. 830-1998
* IEEE std. 1362-1998

**VALIDACIÓN DE LOS REQUERIMIENTOS**

Verificar por medio de listas de chequeo si los requerimientos están bien estructurados. Las listas de chequeo que utilizaremos serán las de CxOne.

**Conducto de las revisiones de los requerimientos**

Se debe revisar constantemente los requerimientos, esto con el fin de que no afecten en el momento de la implementación, también cuando se desean hacer cambios a estos. Se deben revisar detenidamente los errores cometidos.

Este tema esta muy relacionado con la trazabilidad, es decir determinar cual es el origen, el destino.

# **DESCRIPCIÓN GLOBAL**

*En general en esta sección se describe los factores generales que afectan al producto y sus requerimientos, es importante aclarar que en esta sección NO se especifican formalmente los requerimientos, es solo información de fondo que brinda a los lectores una descripción de todo el sistema.*

## **PERSPECTIVA DEL PRODUCTO**

*Esta sección describe el contexto y origen del producto que va a ser especificado en este SRS. La ilustración 1 muestra los diferentes tipos de productos de software que pueden ser desarrollados y, dependiendo de dicho tipo de producto se le debe dar un enfoque diferente a esta sección.*

Ilustración 4: Tipos de productos de software

### INTERFACES CON EL SISTEMA

*En esta sección se deben definir y especificar todas las interfaces que usara el sistema para comunicarse con otros. Un problema común en esta sección es que se define el motor de base de datos como otro sistema y esto no es así, dado que dicho motor también hará parte de nuestro sistema. Para cada tipo de producto esta sección se especifica como sigue.*

* *Producto que pertenece a una familia de productos: Especificar como los demás productos de la familia pueden comunicarse con el producto que se está desarrollando.*
* *Producto remplaza a un sistema actual: Por lo general, cuando un sistema es reemplazado, algunos componentes se deben reutilizar, así que los dos sistemas (nuevo y antiguo) deben definir su forma de comunicación. Esto no es una tarea sencilla ya que en ocasiones el sistema antiguo fue programado en otro lenguaje así que se deben implementar estrategias como CORBA (Common Object Request Broker Architecture) [http://www.omg.org/gettingstarted/corbafaq. htm], que es una arquitectura que permite que dos aplicaciones en diferentes lenguajes puedan interactuar.*
* *Componente de un sistema: En este tipo de proyectos donde un componente es un incremento más para llegar al sistema total, es totalmente necesario definir y especificar las interfaces con los demás componentes (módulos, paquetes, etc), así como también se debe definir mediante que interfaces los demás componentes se comunicaran con el componente a desarrollar.*
* *Producto totalmente nuevo: por lo general los productos nuevos no interactúan con otros sistemas, pero se debe especificar las interfaces del producto a desarrollar para el caso que nuevos componentes interactúen con el sistema. Vale aclarar, que por lo general cuando se realizan sistemas transaccionales nuevos, dichos productos si interactúan con otros sistemas como entidades financieras o administradoras estatales, por lo tanto, para interacción con entidades financieras se deben manejar estándares de seguridad como el ISO 17799[decreto 1465] que garantizan la seguridad necesaria en el proceso de remisión y recepción de la información.*

### INTERFACES CON EL USUARIO

*En esta sección se especifican las características lógicas de cada interfaz entre el producto de software y sus usuarios [3]. Por otra parte, en esta sección también se deben especificar los aspectos de optimización de interfaces, en resumen, esto significa que para hacer medibles los requerimientos de interfaz se debe definir métricas para calificar la usabilidad de una interfaz en particular. Por ejemplo, para hacer medible la interfaz gráfica de usuario asociada a la operación “Registrar Usuario” se puede medir el tiempo necesario en entrenamiento para que el usuario pueda realizar dicha operación mediante la interfaz gráfica [4].*

*A continuación se presenta un ejemplo de las interfaces de usuario genéricas usadas en la mayoría de proyectos de software actuales:*

Ilustración 5: Interfaces con el usuario

*Cómo se ve en la ilustración 5, además de definir las interfaces que usara el sistema, también se deben dar especificaciones técnicas necesarias para que cada interfaz permita este en la capacidad de permitir la interacción entre sistema y usuario. Estas especificaciones técnicas no solo hacen referencia a capacidades de computo sino también, como estas interfaces interactúan con el usuario en caso de algún tipo de incapacidad.*

### INTERFACES CON EL HARDWARE

*En esta sección se especifica como el software va a interactuar con componentes hardware, en general, se definen y especifican puertos de comunicación, conjuntos de instrucciones, dispositivos soportados y protocolos utilizados [3] y [4]. A continuación se muestra un ejemplo de protocolos y puertos utilizados para una aplicación distribuida:*

* *Protocolo de comunicación TCP/IP: (Justificar porque el uso de este protocolo) Ejemplo: Se usará el protocolo de red TCP/IP ya que es un mecanismo de transporte confiable y además es orientado a conexión, evitando recepción de paquetes incompletos o dañados. No se usará protocolo UDP ya que este no es confiable en flujos de paquetes en tiempo real y el porcentaje de pérdida de datagramas es muy alto [5].*
* *Puertos de red TCP: (Para la comunicación de una aplicación distribuida se deben especificar los puertos de comunicación para configurar los firewalls en el lugar de trabajo) Ejemplo: La aplicación utilizara el puerto TCP 7777, mayor a 1024 para evitar conflictos con otras aplicaciones.*
* *Cables y conexiones: (Definir como los diferentes ordenadores de la aplicación realizarán su interacción) Ejemplo: para establecer la conexión entre los equipos se utilizara cable UTP conectados a un dispositivo de interconexión tales como hubs, switches y routers.*

### INTERFACES CON EL SOFTWARE

*En esta sección se especifican como se comunicara el sistema a desarrollar con otros productos de software como por ejemplo sistemas gestores de bases de datos (DBMS), sistema operativo, maquinas virtuales, etc [3]. Para cada uno de estos productos de software se debe especificar nombre, número de versión y fuente; dado que con base a este documento se realizara la implementación del producto, es necesario especificar la documentación de las interfaces que se utilizarán. En esta sección también se debe justificar porque utilizar en el sistema a desarrollar otros productos de software [4]. A continuación se presenta una tabla ejemplo para un desarrollo con recursos gratuitos exceptuando el sistema operativo (Microsoft Windows XP Professional).*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Producto de Software | Windows | phpMyAdmin | MySQL | JDBC | JVM |
| Descripción | La aplicación será desarrollada para la plataforma Windows que es una familia de sistemas operativos basados en una interfaz gráfica de usuario que aplican el paradigma de ventanas.[8] | Este DBMS es una herramienta escrita en PHP que permite manejar la base de datos seleccionada (MySQL). Además, es una herramienta robusta que brinda diferentes funcionalidades como lo son: CRUD de tablas, ejecución de sentencias en SQL, administrar privilegios, entre otras.[9] | Este sistema de gestión de bases de datos soporta la realización de transacciones en simultánea, multiusuario y multihilo; además, cuenta con la GNU GPL. [10] | Es un API que permite llevar a cabo operaciones sobre las bases de datos utilizando Java como lenguaje de programación; es independiente del sistema operativo y de la base de datos en la cual se utilice, emplea el lenguaje de query SQL.[11] | La máquina virtual de Java es un programa instalable en el sistema operativo que permite la interpretación y la ejecución de código de java. |
| Propósito de Uso | Asegurar que la mayoría de los usuarios finales puedan utilizar el producto sin inconvenientes, pues este sistema operativo es el más difundido y usado en la actualidad. Además, el equipo de desarrollo tiene un mayor y fácil acceso a este. | Realizar la administración de la base de datos con facilidad, utilizando páginas Web, Internet, sentencias SQL, lo cual es proveído por este DBMS. | Utilizar una base de datos segura que soporte múltiples consultas, proporcione una capacidad de respuesta y almacenamiento eficiente y además, que asegure la integridad de la información. | Acceder a la base de datos por medio de un manejador de conexiones que implemente las interfaces en lenguaje de Java, debido a que la aplicación requiere realizar operaciones de consulta y actualización constantemente a la base de datos | Puesto que el lenguaje de programación que se usará para el desarrollo de la aplicación, que es Java, no es un lenguaje de bajo nivel o de máquina es necesario usar un intérprete que permita la correcta ejecución de estas instrucciones. |
| Versión | Versiones basadas en la tecnología NT | Versión 2.8.0.1 | Versión 4.1.18 | Versión 4.0 | Versión 1.6 |
| Fuente | Microsoft Corporation  Ayuda:  http://windowshelp.microsoft.com/Windows/es-ES/default.mspx | The pdhpMyAdmin project  Documentación: http://www.phpmyadmin.net/documentation/ | MySQL AB  Documentación:  http://dev.mysql.com/doc/ | Sun Microsystems  Documentación:  http://java.sun.com/j2se/1.4.2/docs/guide/jdbc/index.html | Sun Microsystems  Documentación:  http://java.sun.com/products/archive/jdk/1.1.8\_009/jre/index.html |
| Comentarios Adicionales | La aplicación podría funcionar en otros sistemas operativos que soporten el JRE versión 1.6. | Se utilizará leguaje SQL para realizar el CRUD de las tablas en la base de datos. | El equipo cuenta con más de un servidor que tiene este gestor de base de datos, lo que brinda facilidad de acceso y permite al equipo de desarrollo afrontar contratiempos de disponibilidad de la base de datos. | Puesto que JDBC es independiente del S.O, permite que la aplicación pueda funcionar en otras plataformas diferentes a Windows | Una gran ventaja de la JVM es la portabilidad que le brinda a Java, permitiendo la ejecución de programas desarrollados en este lenguaje en S.O que soporten la máquina virtual. |

Tabla 3: Interfaces con el Software

### INTERFACES DE COMUNICACIÓN

*En esta sección se especifica todas las interfaces de comunicación utilizadas tales como protocolos de red local etc [4]. Por lo general, cuando las aplicaciones hacen uso de diferentes computadores, se utiliza TCP/IP mediante una LAN o una WLAN, para esto se aconseja hacer referencia al RFC de TCP, pero cuando el producto de software utiliza comunicación por serial y los mensajes son únicos para el sistema es necesario especificar que significa cada bit de la trama.*

### RESTRICCIONES

*En resumen esta sección específica características especiales de tarjeta de video, memoria volátil, dispositivos de almacenamiento secundario, etc [4]. Para esta sección se recomienda hacer una sumatoria de los requerimientos de memoria de cada uno de los productos de software usados para el desarrollo del sistema, el siguiente ejemplo está relacionado con los productos utilizados en la Tabla 3:*

* Máquina Virtual de Java: Para la instalación del JRE v. 1.6 se requieren 80 MB de disco duro libre y 64 MB de RAM.
* PhpMyAdmin: Para la instalación del controlador de la base de datos, la cual debe encontrarse en la misma máquina de la base de datos, es necesario contar con al menos 128 MB de RAM, esto para permitir la utilización de todas las aplicaciones necesarias durante la ejecución de la aplicación.

*Existen otros tipos de restricciones memoria para aplicaciones web como las del explorador web entre otras. Para las aplicaciones tipo stand-alone las restricciones de memoria están sujetas al diseño del sistema ya que es la única forma de conocer su uso de recursos*

###### Restricciones de memoria

Las características minimas que debe tener un equipo donde se establecera

###### Restricciones de almacenamiento

###### Restricciones de equipos

###### Restricciones de tiempo

###### Restricciones de personal

### OPERACIONES

*Esta sección debe especificar las operaciones especiales requeridas por el usuario tales como:*

Ilustración 6: Operaciones

### REQUERIMIENTOS DE ADAPTACIÓN DEL SITIO

*Debe definir los requerimientos para adaptar el software a un sitio dado o a un modo operacional. Además debe especificar las características que se deben modificar para adaptar el software a una instalación particular.*

## **FUNCIONES DEL PRODUCTO**

* + 1. **Funciones Principales**
    2. **Stakeholder**
       1. **Actores**
       2. **Usuarios**
    3. **Componente grafico**
    4. **Diagrama y casos de uso**
    5. **Matriz de relaciones**

*Esta sección es una descripción general de las funciones que debe cumplir el producto de software [4]. El listado de funciones debe proveer la especificación clara de cada una, con el fin de asegurar que cualquier persona que lo lea, ya sea el cliente o desarrolladores puedan comprenderlo. Es aconsejable utilizar diagramas, gráficas, entre otros medios visuales que permitan comprender la relación lógica entre las funciones. Por ejemplo, un diagrama de casos de uso con su respectiva documentación.*

Ilustración 7: Tips para definir funciones del producto

*En la ilustración 7 se presentan las ideas claves que se deben tener en cuenta para definir y describir las funciones del producto.*

## **CARACTERÍSTICAS DEL USUARIO FINAL**

*Debe Identificar las diferentes clases de usuarios que usted prevea utilizaran la aplicación. Las diferentes clases de usuarios pueden ser diferenciadas basados en la frecuencia del uso, los diferentes roles, la experiencia técnica, los niveles de seguridad o de privilegios, el nivel de estudios, o de experiencia.*

*Además en general debe describir las características pertinentes de cada clase de usuario, las cuales se ilustran en la ilustración 8 y luego se describen en la Tabla 4 [1].*

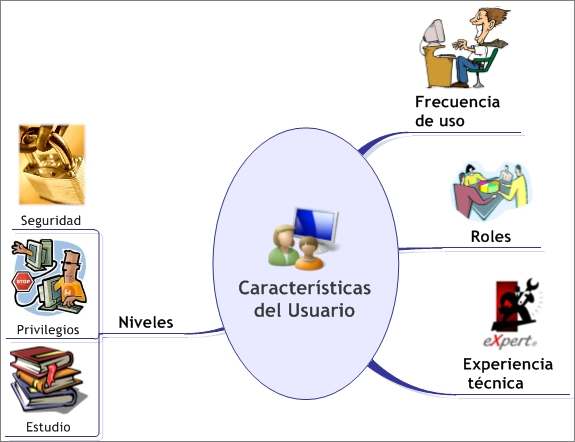
**

Ilustración 8: Características del Usuario

|  |  |
| --- | --- |
| Características del Usuario | Descripción |
| Nivel de Seguridad o de Privilegios | Listar todos los privilegios o funcionalidades a las que tiene acceso el usuario. |
| Roles | Los diferentes roles que puede desempeñar el usuario, como por ejemplo: jugador, visitante, administrador, etc. Especificando las características especificas de cada uno de los roles. |
| Nivel de Estudios o Experiencia Técnica | El nivel de estudio o de experiencia técnica relacionada con el uso de la aplicación. |
| Frecuencia de Uso | Periodicidad con la que el usuario utiliza la aplicación. |

Tabla 4: Descripción de las Características del Usuario

## **RESTRICCIONES**

*Debe especificar las características o restricciones del sistema entre las cuales se encuentran las descritas en la ilustracion9:*

Ilustración 9: Restricciones

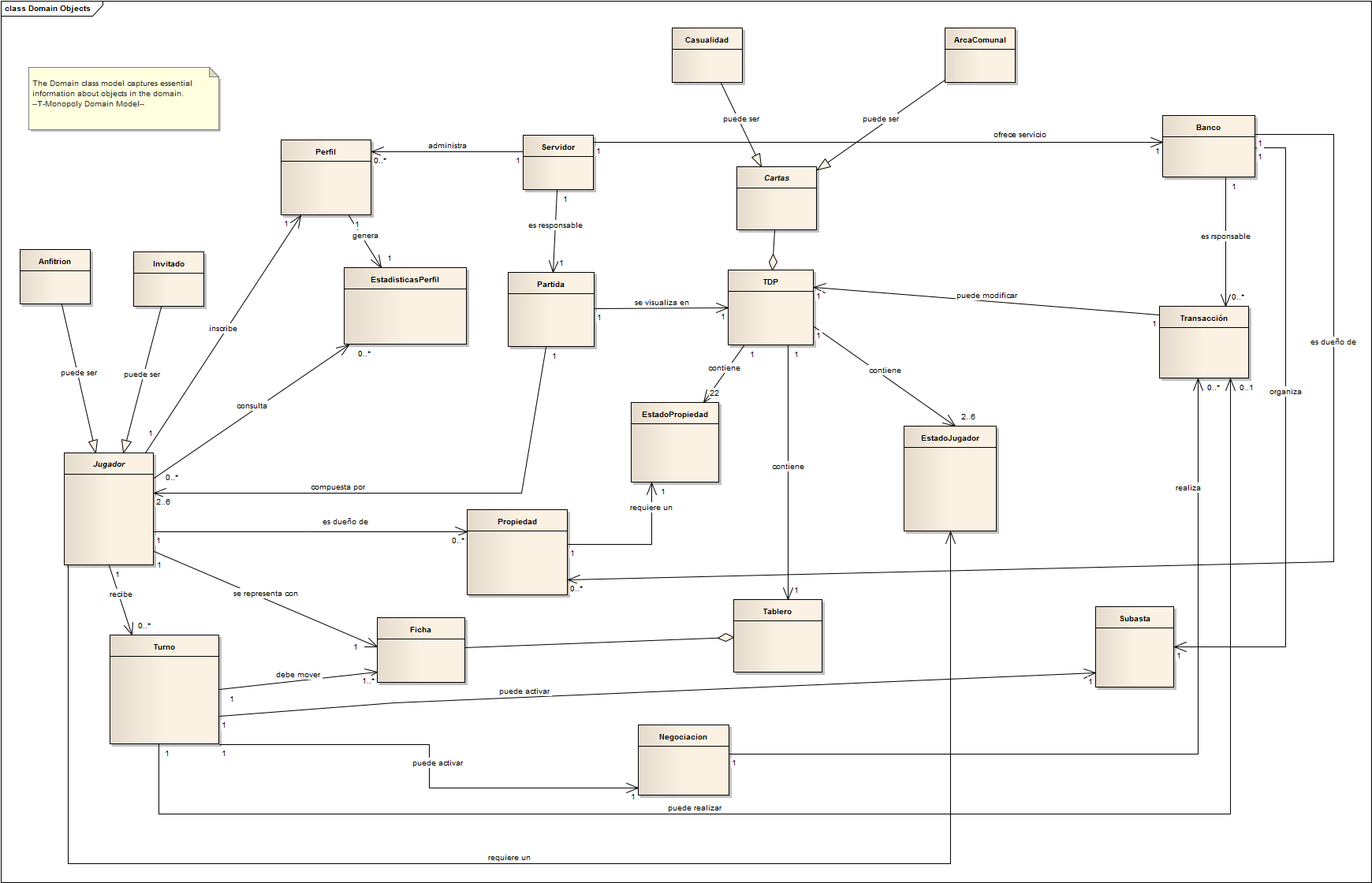
*Además se debe proporcionar una descripción general de cualquier aspecto que limitara las opciones de los diseñadores. Algunas de estas se muestran en la ilustración 10 [1][3]:*

Ilustración 10: Limitaciones

## **MODELO DEL DOMINIO**

Esta sección presenta el modelo del dominio de T-Monopoly® junto con su documentación

* + 1. **Diagrama**

****

***Ilustración 12****:* ***Modelo del dominio***

* + 1. **Clases conceptuales**
    2. **Relaciones**

## **SUPOSICIONES Y DEPENDENCIAS**

*Lista de factores que afectan los requerimientos:*

*Suposiciones:*

*Se deben listar todas aquellas suposiciones que puedan llegar a afectar los requerimientos indicados en la sección 3. Estos pueden incluir componentes comerciales o de terceras personas que usted planea utilizar. Tenga en cuenta que el proyecto podría afectarse si estas suposiciones son incorrectas, no se comparten, o se cambian [1].*

Ilustración 12: Suposiciones

*Dependencias:*

Ilustración 13: Dependencias [1]

## **DISTRIBUCIÓN DE REQUERIMIENTOS**

Para realizar la clasificación de los requerimientos tuvimos en cuenta la pirámide de requerimientos, la cual a partir de los recursos y las características comunes permite dividir los requerimientos en diferentes tipos, como por ejemplo:

http://www.ibmpressbooks.com/articles/article.asp?p=1152528&seqNum=2

A partir de los requerimientos antes nombrados acorde a las diferentes características en común, se plantea la siguiente pirámide denominada pirámide de requerimientos, en la imagen a continuación se muestra claramente como se dividen los diferentes aspectos a tener en cuenta para la división de los requerimientos :



http://ptgmedia.pearsoncmg.com/images/chap1\_9780321383006/elementLinks/01fig01.gif

Basándonos en el modelo anterior hemos decidido realizar la clasificación de los requerimientos en este SRS para el desarrollo del juego T- Monopoly, ya que involucra las necesidades de cada uno de los agentes involucrados relacionándolos con las actividades encaminadas para lograr la realización de cada una de estas tareas, de esta forma el proceso de desarrollo y diseño de la arquitectura será mucho más manejable, por la claridad que se tendría de cada uno de los procesos y actividades que se buscan realizar y quién las realizará, permitiendo que el diagrama y posterior código sea consistente con los requerimientos planteados y con ello con las necesidades del cliente final.

Otra aspecto importante de esta clasificación es que permite relacionar los casos de uso de una forma clara y concisa, facilitando de esta forma la identificación de los casos de uso que faltan y de los que sobran acorde a las necesidades y características identificadas para el juego T-Monopoly.

Para generar el diagrama correspondiente a la pirámide de requerimientos de Alimnova® partimos de las necesidades del cliente, Miguel Torres, que también son nuestras restricciones y base para el desarrollo del juego, las cuales son: Interfaz gráfica fuerte, Persistencia y Arquitectura Cliente / Servidor.

el proyecto.

En la gráfica anterior identificamos como la base las restricciones dadas por el cliente, las cuales son: Interfaz gráfica fuerte, Persistencia y Arquitectura Cliente / Servidor.

A partir de estas tres grandes bases se despliegan las características correspondientes a cada restricción, de forma que a cada restricción la relacionamos con su labor en el juego, el desempeño que va a tener en este ó las actividades que esta desarrolla bien sea durante el juego, respecto o a la funcionalidad y la relación con la visualización que tiene el usuario del juego ó para poder crear y mantener el juego como tal, a partir de estas posibles relaciones que se puede tener en el juego generamos las posibles características que tendrían cada una de las restricciones planteadas.

Luego de tener las características de estas relaciones de esta forma podemos identificar los casos de uso asociados a cada uno, las relaciones que pueden tener y cómo se van a desarrollar dentro del sistema y el desarrollo del juego como tal, respecto tanto a funcionalidad del juego, la forma en que el usuario ve el juego, la parte correspondiente al sistema, de crear conexiones, partidas, gestionar dinero, y demás aspectos correspondientes con la lógica del juego, cada componente de esta forma tendrá un caso de uso asociado a partir de la tarea que realice en el juego y que contribuya a asegurar el buen funcionamiento del mismo.

Teniendo los casos de uso podemos identificar los escenarios relacionados con cada uno de ellos y generar las pruebas para desarrollar la evaluación de cada uno de ellos, facilitando así el proceso de evaluación también, puesto que a partir del diagrama generado a partir de estas relaciones podemos identificar las necesidades que debe suplir y cómo lo hace.

*[REFERENCIA: IBM REQUERIMENTS MANAGEMENT USING IBM RATIONAL REQUISITE PRO]*

*http://www.ibm.com/developerworks/rational/library/04/r-3217/index.html*

http://books.google.com.co/books?id=3XSL7kS8GEgC&pg=PA80&dq=REQUIREMENT+PYRAMID+SOFTWARE&ei=tOWaS-YWhPLIBPmyyOMK&cd=3#v=onepage&q=REQUIREMENT%20PYRAMID%20SOFTWARE&f=false

<http://books.google.com.co/books?id=h4pPpXp-xrEC&pg=PA230&dq=REQUIREMENT+PYRAMID+SOFTWARE&ei=tOWaS-YWhPLIBPmyyOMK&cd=1#v=onepage&q=&f=false>

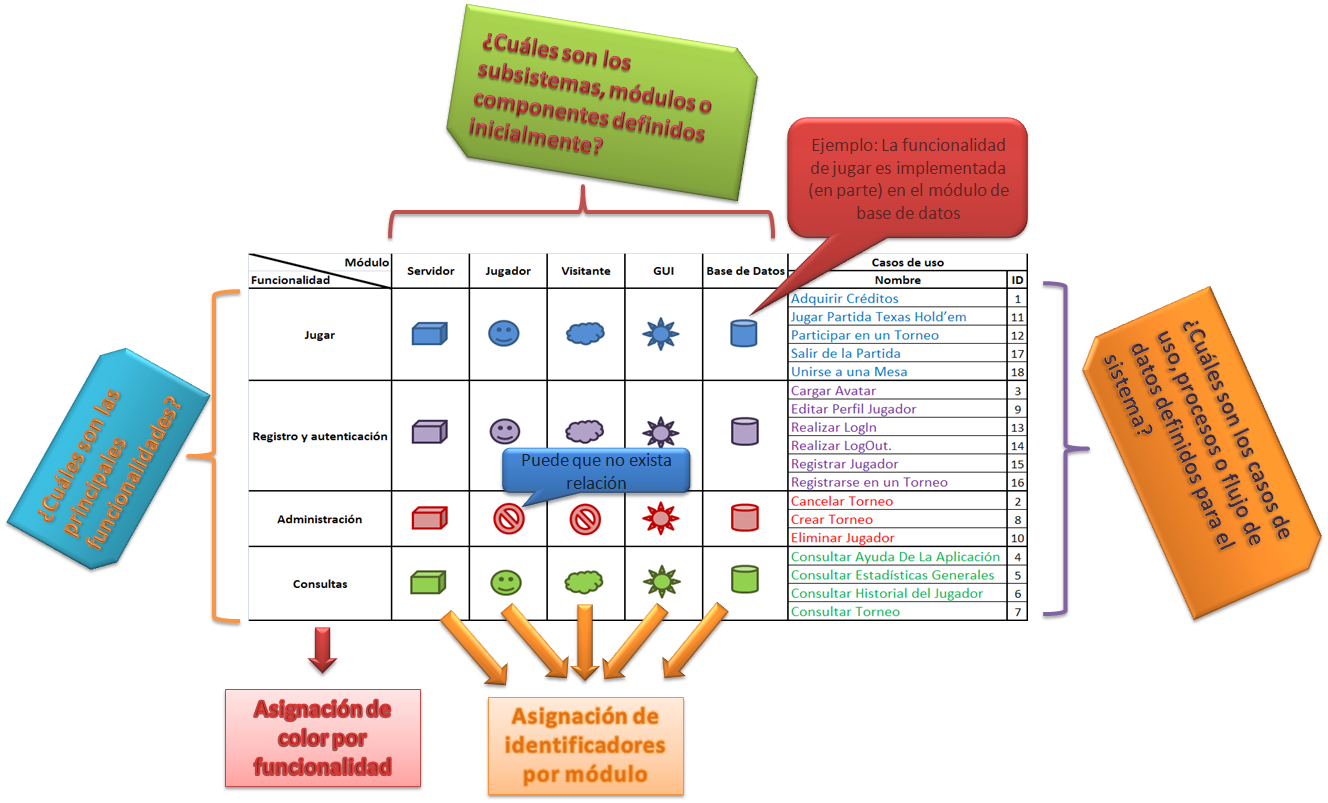
*http://books.google.com.co/books?id=HVvwMGGQqpoC&pg=PA145&dq=REQUIREMENT+PYRAMID&ei=mOWaS9XNJpT4zATBs4T9Cg&cd=1#v=onepage&q=REQUIREMENT%20PYRAMID&f=false*

*<http://books.google.com.co/books?id=Mjn53zPoDQIC&pg=PA212&dq=REQUIREMENT+PYRAMID&ei=mOWaS9XNJpT4zATBs4T9Cg&cd=3#v=onepage&q=REQUIREMENT%20PYRAMID&f=false>*

[*http://books.google.com.co/books?id=MSYLHYhoU3cC&pg=PA105&dq=NEED+DIAGRAM&ei=LeaaS\_-RFovKzgS0lvWMCw&cd=1#v=onepage&q=NEED%20DIAGRAM&f=false*](http://books.google.com.co/books?id=MSYLHYhoU3cC&pg=PA105&dq=NEED+DIAGRAM&ei=LeaaS_-RFovKzgS0lvWMCw&cd=1#v=onepage&q=NEED%20DIAGRAM&f=false)

[*http://books.google.com.co/books?id=e7ZhVD3JejAC&printsec=frontcover&dq=REQUIREMENTS&ei=yuiaS42PHoPUzQSH9aj8DQ&cd=2#v=onepage&q=&f=false*](http://books.google.com.co/books?id=e7ZhVD3JejAC&printsec=frontcover&dq=REQUIREMENTS&ei=yuiaS42PHoPUzQSH9aj8DQ&cd=2#v=onepage&q=&f=false)

*http://books.google.com.co/books?id=Bg4NAQAAIAAJ&pg=PA6&dq=DIAGRAMAS+DE+NECESIDADES&ei=FuaaS7ajBpT4zATBs4T9Cg&cd=1#v=onepage&q=DIAGRAMAS%20DE%20NECESIDADES&f=false*

Ilustración 14: Distribución de requerimientos

# **REQUERIMIENTOS ESPECÍFICOS**

*Esta sección es en cierto sentido la más importante del proyecto en cuanto a diseño e implementación se refiere. La idea principal de todo el documento, es la de realizar una conexión, un vínculo entre el cliente final y los desarrolladores de la aplicación o el sistema y esta sección es la encargada de esa comunicación.*

*La especificación de requerimientos busca trasladar las necesidades reales del usuario a un lenguaje más técnico que facilite su construcción, sin embargo es importante mantener un cierto nivel de descripción y lenguaje, para que el cliente no se pierda en tecnicismos y de esta forma hacerlo parte activa de todo el proceso (diseño y construcción).*

*Para hacer de este documento una parte valiosa en el proceso de construcción del sistema, todos los requerimientos que se ubiquen en esta sección deben cumplir con las siguientes características son [12]:*

* *Atómico: TODOS los requerimientos debe ser atómicos, no compuestos, por lo tanto deben tener solo un propósito y su funcionalidad no debe poder ser divida en más de una parte*
* *Correctos: los requerimientos que se incluyan en el documento deben ser solo los que tienen que ser incluidos en el sistema según el análisis previo y las entrevistas con los clientes y demás stakeholders. Requerimientos innecesarios o con enfoques a tecnologías o gustos de un solo usuario, hacen más extenso el diseño. atrasan la implementación y al final prueban ser inútiles para el sistema en general.*
* *No Ambiguos: los requerimientos finales deben tener solo una interpretación, no importa quién los lea, para facilitar el cumplimiento de esta característica es el diseño de términos que se refieran a una función en particular, por ejemplo palabras que se usen día a día en el negocio o en la empresa y que puedan entenderse con facilidad. Sin embargo el uso de estas palabras requieren la inclusión en el glosario de los documentos (SPMP, SRS, SDD).*
* *Completos: para que un requerimiento sea completo es preciso que incluya toda la información necesario para su perfecto entendimiento, esto es descripción, razón y criterio de medición, entre otros (para más información remitirse a la . La idea principal de que un requerimiento sea completo, es mantener al lector enfocado en un solo documento, y no que tenga que mirar en otro lado para poder entender lo que está escrito.*
* *Consistente: cada uno de los requerimientos debe estar pensado para que no contradiga otro requerimiento u otra sección de alguno de los documentos, de esta forma a la hora de implementar el sistema no se presentarán errores de redundancia o conflictos de intereses. Además es importante utilizar el mismo tipo de lenguaje entre requerimientos (ver No Ambiguos).*
* *Importancia: los requerimientos incluidos en el documento deben estar clasificados según el orden de importancia que ocupan dentro del sistema, de esta forma es más fácil para los programadores y desarrolladores enfocarse en las funcionalidades más críticas de la aplicación. Por ejemplo “Agregar Jugador” no es tan importante si “Crear un Juego” no funciona.*
* *Verificable: todos los requerimientos deben tener una forma con la cual se pueda saber si se están cumpliendo con los objetivos, o mejor si la funcionalidad de dicho requerimiento está presente en la implementación. Algunos de los métodos para verificar si un requerimiento es verificable son: Inspección, Análisis, Demostración y Prueba [13].*
* *Modificable: cualquier cambio a los requerimientos debe poder hacerse de manera sencilla, manteniendo la estabilidad del sistema y al mismo tiempo su estructura y estilo. La redundancia de requerimientos no permite su fácil ubicación y mucho menos su cambio.*
* *Trazable: un requerimiento es trazable si su origen es claro y además si su relación con otro tipo de requerimientos es fácilmente mapeable. Cuando un sistema entra en su fase de mantenimiento, la trazabilidad de los requerimientos es fundamental para llevar a cabo una fácil transición.*
* *Asociados a Versión: los requerimientos deben estar ordenados de acuerdo a la evolución y deben estar relacionados con el producto que satisfacen*
* *No Redundante: los requerimientos no deben estar en más de una parte del documento ni deben repetirse.*
* *Precisos: no dejan lugar a malas interpretaciones ni necesitan ser explicados, son concisos. Si es necesario (números) se debe hacer inclusión de cifras y posiciones decimales.*

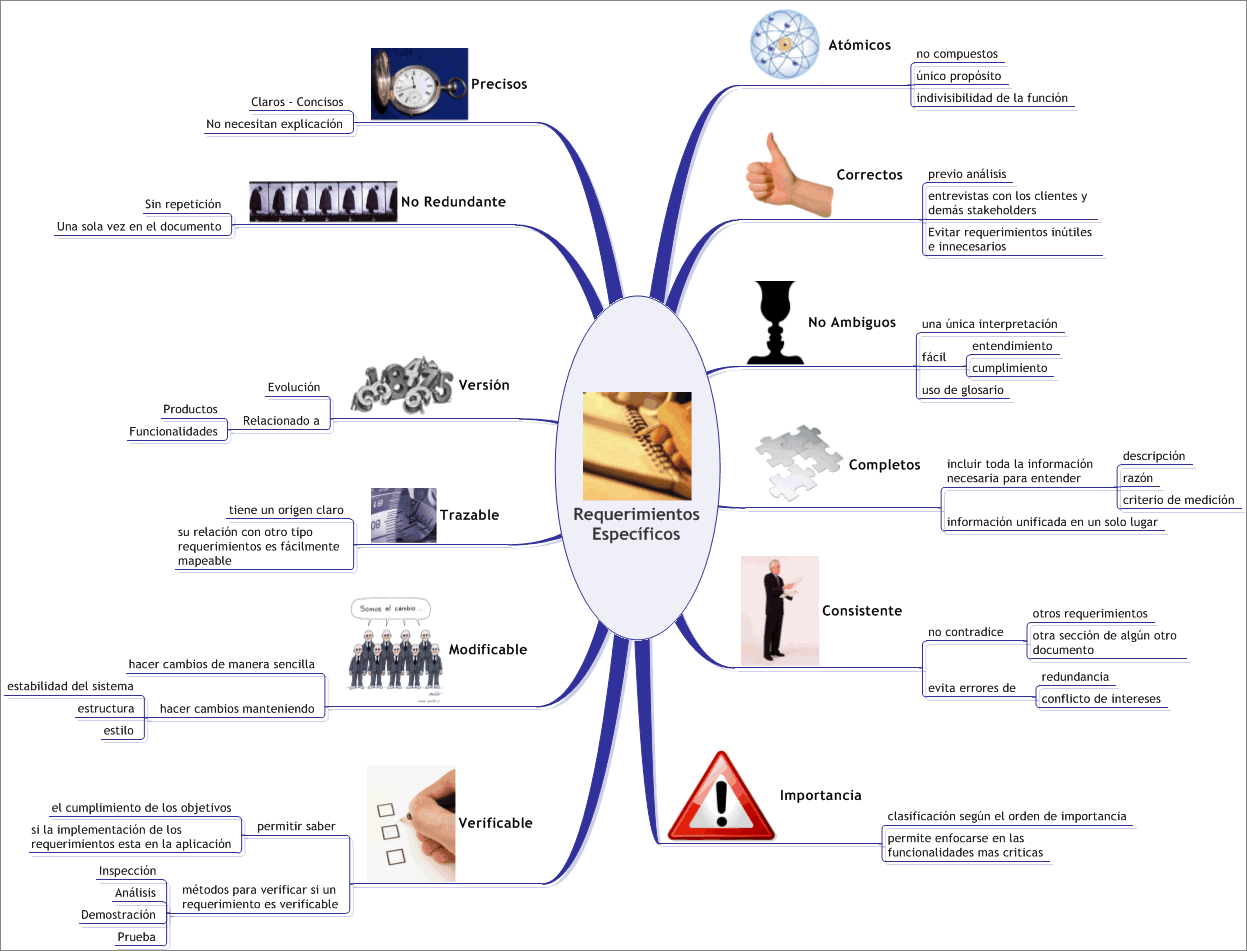
**

Ilustración 15: Características de los Requerimientos

*La basada en la plantilla de Volere [14] presenta un posible formato que permite especificar los requerimientos funcionales de manera que se cumpla con las características anteriores.*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *# Requerimiento* |  | *Tipo de Requerimiento* | |  | *Casos de Uso Asociados* | |  |
| *Descripción* |  | | | | | | |
| *Razón* |  | | | | | | |
| *Autor* |  | | | | | | |
| *Criterio de medición* |  | | | | | | |
| *Prioridad* |  | | *Módulo Asociado* | | |  | |
| *Versión* |  | | *Fecha* | | |  | |

Tabla 7: Documentación de Requerimientos

Ilustración 16: Documentación de Requerimientos

## **REQUERIMIENTOS DE INTERFACES EXTERNAS**

Las siguientes secciones se encuentran estrechamente relacionadas con la sección 2.1, la explicación de estas, busca simplemente profundizar un poco más en el contenido que debe tenerse en cuenta en el momento de llenar esta plantilla y definir los requerimientos de Interfaces Externas.

### INTERFACES CON EL USUARIO

En esta sección se debe explicar la forma en que el sistema o que la aplicación permitirá la comunicación con el usuario o el cliente final. Esta comunicación incluye el ingreso de datos al sistema, la navegación por ventanas, la forma en que se muestran las interfaces (si las hay) y los diferentes dispositivos del computador en el que se va a utilizar el sistema en construcción.

Las interfaces deben contener la información lógica, por ejemplo formatos de pantalla requeridos (1024\*768, 800\*600,600\*400), distribución de elementos en la pantalla o si tiene accesos rápidos con el teclado, por dar algunos ejemplos [12].

Algunas interfaces que se deben tener en cuenta son:

Ilustración 17: Interfaces con el Usurario

### INTERFACES CON EL HARDWARE

Puesto que el sistema que se planea desarrollar usualmente debe estar en capacidad de compartir datos e información con otros computadores y dispositivos (ejemplo dispositivos móviles), es necesario tener en cuenta la forma en que los componentes de software (aplicación o sistema) se comunicaran con los componentes de hardware de los otros dispositivos.

*Las características del sistema a nivel de hardware que se deben tener en cuenta para el desarrollo, se enfocan en cómo se va a llevar a cabo la comunicación entre las máquinas de los usuarios participantes, algunas de estas interfaces son:*

Ilustración 18: Interfaces de Hardware

### INTERFACES CON EL SOFTWARE

Para la ejecución del sistema apropiadamente es necesario tener algunos requerimientos mínimos de Software, esto es versión del sistema operativo, servidores de bases de datos y en general todos los productos de software que permitan la correcta utilización del sistema desarrollado. La muestra un formato que puede usarse para explicar estas interfaces.



Ilustración 19: Interfaces con el Software

### INTERFACES DE COMUNICACIONES

Las interfaces de comunicación son básicamente los métodos como se interconectan las diferentes aplicaciones en las máquinas en donde se ejecutan (nuevamente si la aplicación es distribuida), estas interfaces incluyen el tipo de red que se debe montar para la conexión de computadores (LAN, WAN) y los protocolos de comunicación usados (TCP). Para apoyar la información escrita en esta sección, se aconseja explicar cierta parte en la sección 3.1.2

## **CARACTERÍSTICAS DEL PRODUCTO DE SOFTWARE**

*Los requerimientos funcionales definen las acciones fundamentales que deben encontrarse y ser cumplidas en el sistema, aceptando entradas, procesando y generando las salidas. Éstos generalmente se redactan de la siguiente forma: "El sistema debe…."*

Para esta sección es aconsejable utilizar la con el fin de hacer de los requerimientos lo más entendibles y completos posibles. Además se sugiere realizar el acoplamiento con los casos de uso, permitiendo una mayor trazabilidad y la capacidad de modificar requerimientos conociendo el impacto en todo el sistema.

Puesto que esta sección es la más importante del SRS, en ella se enumeran y se explican todas las características que debe poseer el sistema a implementar, y posiblemente la más extensa, es aconsejable tratar de dividirla por módulos, por funcionalidades o por cumplimiento de casos de uso.

*A continuación se presenta una división basada en el proyecto 7 Texas Poker [15] en el cual se decidió fraccionar esta sección del por funcionalidades del sistema:*

Ilustración 20: División por Funcionalidades

* + 1. Funcionalidad 1: Jugar

*Uso de la para la documentación de los requerimientos asociados a esta funcionalidad.*

* + 1. Funcionalidad 2: Registro y Autenticación

*Uso de la para la documentación de los requerimientos asociados a esta funcionalidad.*

* + 1. Funcionalidad 3: Administración

*Uso de la para la documentación de los requerimientos asociados a esta funcionalidad.*

* + 1. Funcionalidad 4: Consultas

*Uso de la para la documentación de los requerimientos asociados a esta funcionalidad.*

* + 1. Cruce de funcionalidades

*Uso de la para documentar los requerimientos que no pueden incluirse en alguna de las funcionalidades principales.*

*Tanto para esta sección como para la siguiente es muy importante tener en cuenta que todos los requerimientos deben tener su identificador y si se desea el caso de uso al que está asociado, esto con el fin de cumplir con la característica de trazabilidad y de esta forma en caso de modificación o de implementación facilitar la ubicación del requerimiento*.

## **REQUERIMIENTOS DE DESEMPEÑO**

*Una vez se han establecido los requerimientos funcionales en la sección anterior, es hora de incluir los requerimientos de desempeño. En este punto se realiza la división en requerimientos estáticos que incluyen la cantidad de usuarios concurrentes o simultáneos que el sistema puede y debe soportar, contando las terminales o equipos utilizados y conectados al mismo tiempo, y en requerimientos dinámicos que pueden incluir el número de transacciones que debe soportar el sistema dependiendo de la funcionalidad y los tiempos de respuesta del sistema ante las peticiones hechas por los usuarios entre otros.*

*Es importante darle criterios de medición a estos requerimientos de desempeño, de forma que en las fases de verificación y validación se pueda comprobar que el desempeño del sistema, basado en los requerimientos de esta sección, cumpla con las necesidades del cliente o con las especificaciones finales acordadas en el SPMP. Un ejemplo de cómo escribir este tipo de requerimientos es:*



Ilustración 21: Ejemplo Enunciado Requerimientos

*Otros atributos de calidad (Requerimientos no funcionales) con respecto al desempeño del sistema que deben tenerse en cuenta a la hora de documentar las restricciones de diseño son [16]:*

Ilustración 22: Atributos de Calidad a tener en cuenta

## **RESTRICCIONES DE DISEÑO**

*En este numeral se tienen en cuenta las diferentes limitaciones del proyecto en cuestiones de diseño, los principales tópicos que deben tratarse*:

* *Lenguajes de programación*
* *Herramientas CASE (Análisis y Diseño)*
* *Restricciones de Hardware y Software teniendo en cuenta el paradigma y el lenguaje de programación*
* *Diseño de la arquitectura final del sistema (Cliente-Servidor, Repositorio, N-Tier)*
* *Construcción del modelo de bases de datos*

*Puesto que el documento de diseño que se usa para el proyecto es el SDD, es válido nombrar las restricciones sujetas al diseño e incluir referencia a ese documento para futuras o más completas explicaciones*.

## **ATRIBUTOS DEL SISTEMA DE SOFTWARE (NO FUNCIONALES)**

*Para la sección de los requerimientos no funcionales también se puede usar el formato de la . Para la definición de los atributos, al igual que con los requerimientos de desempeño, es necesario incluir los criterios de evaluación medibles. Los principales atributos de calidad que deben ser incluidos en este documento son:*

### CONFIABILIDAD

*Se deben explicar los mecanismos que se van a tener en cuenta para el manejo de la información almacenada en el sistema. También incluye la forma en que la aplicación se va a mantener operativa a lo largo del tiempo especificado en la sección 3.5.2.*

*Ejemplos de este tipo de requerimientos son*:

Ilustración 23: Características de Confiabilidad

### DISPONIBILIDAD

*Porcentaje de tiempo al día o a la semana que el sistema debe funcionar sin necesidad de reiniciarlo. Si por ejemplo el sistema depende o necesita un módulo de comunicación con otras aplicaciones externas, el horario o mejor la disponibilidad de estas últimas también debe tenerse en cuenta para esta sección. Si es necesario que la aplicación o el sistema desarrollado cuenten con un administrador, el tiempo disponible que este* *posea para dedicarse a su rol debe ser incluido al mapear los requerimientos de disponibilidad.*

Ilustración 24: Características de Disponibilidad

### SEGURIDAD

*Este atributo depende en gran cantidad de la información que se maneje en el sistema, usualmente los métodos de seguridad más utilizados son los de permisos para usuarios o creación de cuentas, sin embargo si es necesario también podrían implementarse sistemas de encriptación o registros de acciones ejecutadas por los usuarios loggeados en el sistema. Si la aplicación es susceptible a ataques de Denegación de Servicio, mecanismos como replicación de servidores o balanceadores de cargas también pueden ser tenidos en cuenta*.

Ilustración 25: Características de Seguridad

### MANTENIBILIDAD

*Una característica que hace del producto de software desarrollado fácil de mantener, es su división por funciones y por módulos, de esta forma si se necesita hacer una modificación a un requerimiento o en general a alguna función, no es necesario volver a implementar el sistema desde cero, ni afectar los módulos ya disponibles. Otra característica que se debe tener en cuenta es la de la documentación del código, en caso que la modificación no sea hecha por los desarrolladores originales, con el fin de facilitar el entendimiento de la estructura interna del sistema.*

Ilustración 26: Características de Mantenibilidad

### PORTABILIDAD

*Se debe especificar si el sistema podrá ser migrado a otras plataformas de sistemas operativos o si podrá ser ejecutado en diferentes ambientes de cómputo estos ambientes pueden incluir hardware, software o una combinación de los dos.*

*También se debe tener en cuenta el lenguaje y el compilador utilizados por el equipo desarrollador.*

### 

Ilustración 27: Portabilidad del Sistema

## **REQUERIMIENTOS DE LA BASE DE DATOS**

*Para la especificación de los requerimientos de la Base de Datos, es necesario tener en cuenta varios aspectos, entre estos se encuentran:*

Ilustración 28: Características Bases de Datos

* *Tipos de datos almacenados: dependiendo del motor de base de datos escogidos, es posible tener diferentes tipos de datos, sin embargo al hacer esta sección podrían incluirse los más usados: Char, Varchar, Numeric y Date por mencionar algunos.*
* *Tipo de consultas utilizadas: la forma en que los datos serán accedidos, consultadas e ingresados desde los DAO´s (Data Access Object) para evitar la introducción de sentencias malintencionadas. Los tipos más conocidos son Statement y Prepared Statment.*
* *Indexación de los datos: la eficiencia de las consultas complejas pueden reducirse dependiendo de la forma en que se haga el diseño de la base de datos, una buena forma de mostrar este aspecto es con el diagrama de Entidad Relación*
* *Utilización de Primary Key: al igual que con el aspecto anterior la utilización adecuada de una primary key puede evitar ciclos y además permite y facilita eficiencia en el tiempo de consultas hechas en tiempo real.*
* *Frecuencia de acceso: dependiendo del tipo de sistema que se desea implementar, especificar la frecuencia de acceso a la base de datos incluyendo el número de conexiones abiertas y tener en cuenta el tipo de consulta utilizada, la carga extra que puede producir el manejo de DAO’s puede disminuir, aumentando de esta forma el desempeño en general de la aplicación.*

# **ANEXOS**

REFERENCIAS

[1] Wiegers, Karl. , Software Requirements Specification. Process Goodies 2002, Disponible en http://www.processimpact.com/goodies.shtml

[2] IronWorks, Plantilla SPMP, Segundo Semestre 2007, Pontificia Universidad Javeriana.

[3] Construx Software, Software Requirements Specification CXOne Standard, Construx Software Builder, Inc, Noviembre 2002.

[4] IEEE (Institute of Electrical and Electronics Engineers), IEEE Recommended Practice for Software Requirements Specificacitions, IEEE-SA Standards Board, Junio 1998.

[5] Introduction to TCP/IP [homepage de Internet]. Copyright 1995 PCLT. Disponible en: http://www.yale.edu/pclt/COMM/TCPIP.HTM

[6] Página principal de Windows [homepage de Internet]. ©2007 Microsoft Corporation. [citado 2007 Mar 25]. Disponible en: http://www.microsoft.com/spain/ windows/default.mspx

[7] phpMyAdmin |MySQL Database Administration Tool| [homepage de Internet]. © phpMyAdmin Devel Team. [citado 2007 Mar 25]. Disponible en: http://www. phpmyadmin.net

[8] MySQL AB :: Developer Zone [homepage de Internet]. © 1995-2007 MySQL AB. [citado 2007 Mar 25]. Disponible en: http://dev.mysql.com

[9] Java SE Technologies – Java Database Connectivity (JDBC) [homepage de Internet]. Copyright 1994-2007 Sun Microsystems, Inc. [citado 2007 Mar 25]. Disponible en: http://java.sun.com/javase/technologies/database/index.jsp

[10] Fowler, M. 1996. Analysis patterns: Reusable Object Models, Reading, MA: Addison-Wesley

[11] Larman C. UML Y PATRONES. Una introducción al análisis y diseño orientado a objetos y al proceso unificado. 2nd ed. Aragón DF. Madrid: Pearson Educación. S.A.; 2003.

[12] IEEE (Institute of Electrical and Electronics Engineers), IEEE Guide for Developing System Requirements Specifications, IEEE-SA Standards Board, Abril 1996.

[13] Nuseibeh, B. et al, Requirements Engineering: A Roadmap, [citado 2007 Septiembre 07], Disponible en: http://www.doc.ic.ac.uk/~ban/pubs/sotar.re.pdf

[14] Pagina de Miguel Torres [homepage de Internet]. Bogotá. Ing. Miguel Eduardo Torres Moreno MSc. Copyright - Miguel Torres 2007. [actualizado el 26 Feb 2007; citado 2007 Septiembre 07]. Materias - Ingeniera de Software, Robertson, S. et. At. Mastering the Requirements Process

[15] IronWorks, Especificación de Requerimientos De Software 7 Texas Poker, Primer Semestre 2007, Pontificia Universidad Javeriana

[16] Barbacci, M. et al, Quality Attributes, Software Engineering Institute, Carnegie Mellon University, December 1995