**[Nombre del proyecto]**

[Nombre de la empresa]

**DESCRIPCIÓN DE DISEÑO DE SOFTWARE**

[Fecha]

[Versión del documento]

[Logos]

[Autores]

HISTORIAL DE CAMBIOS

*En esta sección se presenta una tabla que describe la evolución y los cambios que se le realizan al documento desde que se inicia hasta que se haya llegado a la versión base que es entregada al cliente.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Versión** | **Fecha** | **Sección del documento modificada** | **Descripción de cambios (corta)** | **Responsable (S)** |
| *Indica la versión del documento, que depende según la forma de administración de configuraciones seleccionada.* | *Se incluye la fecha en la que fue realizado el cambio del documento.* | *Permite especificar las secciones del documento que fueron modificadas.* | *Es un pequeño resumen de los cambios más relevantes que fueron realizados en la versión* | *Indica las personas del grupo de trabajo que son responsables del o los cambios realizados en el documento.* |

Tabla : Historial de cambios

Contenido

[Lista de Tablas 4](#_Toc180071410)

[Lista de Ilustraciones 5](#_Toc180071411)

[1. Introducción 6](#_Toc180071412)

[1.1 Descripción del Sistema 6](#_Toc180071413)

[1.2 Mapa del Diseño 7](#_Toc180071414)

[1.3 Referencias y Documentos de Apoyo 7](#_Toc180071415)

[1.4 Definiciones, Acrónimos y Abreviaciones 8](#_Toc180071416)

[2. Consideraciones de Diseño 11](#_Toc180071417)

[2.1 Suposiciones 11](#_Toc180071418)

[2.2 Restricciones 11](#_Toc180071419)

[2.3 Entorno del Sistema 12](#_Toc180071420)

[2.4 Metodología de Diseño 13](#_Toc180071421)

[2.5 Riesgos 14](#_Toc180071422)

[3. Arquitectura 15](#_Toc180071423)

[3.1 Apreciación Global 16](#_Toc180071424)

[3.2 Diagrama de Componentes 17](#_Toc180071425)

[3.2.1 Subsistema 1 19](#_Toc180071426)

[3.2.1.1 Componente 1 19](#_Toc180071427)

[3.3 Estrategias de Diseño 19](#_Toc180071428)

[4. Diseño de Alto Nivel 21](#_Toc180071429)

[4.1 Diagrama de despliegue 21](#_Toc180071430)

[4.1.1 Nodo 1 *“Nombre del Nodo”* 23](#_Toc180071431)

[4.1.2 Conector 1 “Nombre del Nodo” 23](#_Toc180071432)

[4.2 Diagrama de Comportamiento e Interacción 23](#_Toc180071433)

[4.2.1.1 Diagrama de Actividad 24](#_Toc180071434)

[4.2.1.1 Diagrama de Secuencia 24](#_Toc180071435)

[5. Diseño de Bajo Nivel 26](#_Toc180071436)

[5.1 Subsistema 1 29](#_Toc180071437)

[5.1.1 Componente 1 29](#_Toc180071438)

[6. Diseño de Interfaces de Usuario 32](#_Toc180071439)

[6.1. Diseño general de la aplicación 32](#_Toc180071440)

[6.2. Árbol de navegabilidad 33](#_Toc180071441)

[Anexos 34](#_Toc180071442)

Lista de Tablas

[Tabla 1: Historial de cambios 1](#_Toc180071443)

[Tabla 2: Ventajas y Desventajas de Estilos Arquitectónicos 16](#_Toc180071444)

[Tabla 3: Subsistema 1 19](#_Toc180071445)

[Tabla 4: Componente 1 19](#_Toc180071446)

[Tabla 5: Estrategias de Diseño 20](#_Toc180071447)

[Tabla 6: Nodo 1 23](#_Toc180071448)

[Tabla 7: Conector 1 23](#_Toc180071449)

[Tabla 8: Documentación de clases 29](#_Toc180071450)

[Tabla 9: Descripción de entradas y salidas 32](#_Toc180071451)

Lista de Ilustraciones

[Ilustración 1: Descripción del sistema 6](#_Toc180071452)

[Ilustración 2: Mapa de Diseño 7](#_Toc180071453)

[Ilustración 3: Suposiciones generales 11](#_Toc180071454)

[Ilustración 4: Restricciones 12](#_Toc180071455)

[Ilustración 5: Entorno del Sistema 13](#_Toc180071456)

[Ilustración 6: Administración de Riesgos 14](#_Toc180071457)

[Ilustración 7: Apreciación Global 16](#_Toc180071458)

[Ilustración 8: Capas Arquitectónicas 17](#_Toc180071459)

[Ilustración 9: Diagrama de componentes 17](#_Toc180071460)

[Ilustración 10: Subsistema Usuario 7 Texas Poker 18](#_Toc180071461)

[Ilustración 11: Estrategias de Diseño 20](#_Toc180071462)

[Ilustración 12: Descripción de Nodos 21](#_Toc180071463)

[Ilustración 13: Descripción Conexiones 22](#_Toc180071464)

[Ilustración 14: Nodo Usuario 7 Texas Poker 22](file:///C:\Documents%20and%20Settings\Juan%20David\Desktop\Proyecto%20Especial\SDD\SDD%201.0.docx#_Toc180071465)

[Ilustración 15: Diagrama de Actividad 24](#_Toc180071466)

[Ilustración 16: Diagrama de secuencia 25](#_Toc180071467)

[Ilustración 17: Diseño de bajo nivel 26](#_Toc180071468)

[Ilustración 18: Niveles de diseño 27](#_Toc180071469)

[Ilustración 19: Ejemplo diseño de bajo nivel 28](#_Toc180071470)

[Ilustración 20: Ejemplo Descripción de Clase 29](#_Toc180071471)

[Ilustración 21: Ejemplo Resumen de Métodos 30](#_Toc180071472)

[Ilustración 22: Ejemplo descripción de método 31](#_Toc180071473)

[Ilustración 23: Diseño general de la aplicación 32](#_Toc180071474)

[Ilustración 24: Ejemplo árbol de navegabilidad 33](file:///C:\Documents%20and%20Settings\Juan%20David\Desktop\Proyecto%20Especial\SDD\SDD%201.0.docx#_Toc180071475)

# Introducción

## Descripción del Sistema

*La ilustración 1 indica cómo realizar la descripción del sistema, en general, en esta sección, se justifica como se satisfacen los requerimientos funcionales y no funcionales con el diseño del sistema escogido. [5]*

Ilustración : Descripción del sistema

## Mapa del Diseño

*En esta sección se especifica mediante que diagramas se definió el diseño del sistema en el presente documento, por lo general, se hacen dos diseños como se explica a continuación:*

* *Diseño de alto nivel donde se incluye:*
* *Estilo arquitectónico*
* *Diagrama de despliegue*
* *Diagrama de componentes*
* *Diagramas de actividad y de secuencia*
* *Diseño de bajo nivel:*
* *Diagramas de clase*
* *Diagrama entidad relación*

*La presente plantilla se orienta a la definición de dichos diseños y además permite la especificación de otro tipo de diseño, este último, es el diseño de interfaces gráficas donde se incluyen el árbol de navegabilidad y la documentación de las interfaces gráficas, la ilustración 2 muestra el diseño al que está orientado la presente plantilla.*

Ilustración : Mapa de Diseño

## Referencias y Documentos de Apoyo

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

## Definiciones, Acrónimos y Abreviaciones

*En esta sección se definen y especifican los acrónimos y definiciones utilizadas a lo largo de este documento, lo ideal, es realizar una recopilación de los acrónimos igualmente usados en los documentos que preceden el SAD (SPMP y SDD), a continuación se muestra una lista usada por lo general en esta sección para la descripción del diseño del software:*

***802.11:*** *El protocolo IEEE 802.11 o WI-FI es un estándar de protocolo de comunicaciones de la IEEE que especifica las normas de funcionamiento en una WLAN.*

***A***

***API - (Aplication Programming Interface)*** *Interfaz de Programación de Aplicaciones, es un conjunto de especificaciones de comunicación entre componentes software.*

***C***

***Commit:*** *Instrucción de manejo de bases de datos que guardad todos los cambios realizados en ella.*

***CRUD – (Create – Retrieve – Update - Delete)***

***D***

***DBMS (DATA BASE MANAGEMENT SYSTEM):*** *sistema dedicado a servir de interfaz entre las bases de datos y las aplicaciones que las utilizan.*

***DET - Data Element Type:*** *Tipo de dato único no recursivo en un ILF o EIF.*

***Disconnect Protection – (Protección de desconexión):*** *En poker en línea, algunas mesas ofrecen una protección para el jugador en caso de que pierda la conexión durante el transcurso de una mano; su mano queda en situación de All-in y tiene derecho a ganar el bote que se había formado hasta el momento en que se desconectó.*

***E***

***ER****: Entidad Relación.*

***F***

***Framework:*** *Estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Puede incluir soporte de programas, librerías y un lenguaje de scripting entre otros.*

***G***

***GNU:*** *Acrónimo recursivo que significa “GNU No es Unix”.*

***GNU GPL:*** *General Public License o licencia pública general.*

***GUI:*** *En el contexto del proceso de interacción persona-ordenador, la interfaz gráfica de usuario (IGU), provee una forma amigable de interacción con un sistema informático.*

***I***

***ILF - Internal Logical Files:*** *Un grupo de datos lógicamente relacionados que reside totalmente dentro de la frontera de la aplicación y es mantenido a través del EI.*

***J***

***JDBC:*** *Java DataBase Connectivity.*

***JRE:*** *Java Runtime Environment.*

***JUnit:*** *Conjunto de librerías utilizadas en programación para hacer tests unitarios de aplicaciones Java.*

***JVM:*** *Java Virtual Machine.*

***L***

***LAN:*** *Red de Área Local (Local Area Network, por sus siglas en Inglés).*

***P***

***PHP:*** *Acrónimo recursivo “Hypertext Preprocessor”.*

***R***

***RET - Record Element Type:*** *Un subgrupo identificable de elementos de datos dentro de un ILF o un EIF.*

***RFC:*** *Request For Comments.*

***RMI: (Java Remote Method Invocation)*** *Mecanismo ofrecido en Java que permite a un procedimiento (método, clase o aplicación) poder ser invocado remotamente.*

***Rollback:*** *Comando de instrucción de bases de datos que permite deshacer los cambios realizados.*

***S***

***SDD:*** *Documento que describe el modelo de diseño del sistema*

***SPMP:*** *Documento que controla un proyecto de software. El SPMP define las actividades, productos de trabajo, indicadores de avance y recursos asignados al proyecto. En el SPMP también están definidos los procedimientos administrativos y convenciones aplicables al proyecto, como el reporte de estado, la administración del riesgo, y la administración de contingencias.*

***SQL:*** *Lenguaje Estructurado de Consultas (Structured Query Language, por sus siglas en Inglés).*

***SRS:*** *Documento que describe el sistema de requerimiento de software*

***U***

***UML:*** *Unified Modelling language, lenguaje de modelado de sistemas de software.*

***UTP:*** *Cable de par trenzado sin apantallar ( Unshielded Twisted Pair).*

***V***

***VAF - Value adjustment factor:*** *Está basado las Características Generales del sistema.*

***W***

***Warnings:*** *Advertencias de un posible error en el código en el momento de compilar el mismo.*

***WLAN:*** *Es un sistema de comunicación de datos inalámbrico, utilizado como alternativa a las redes de área local cableadas o como extensión de éstas. Utiliza tecnología de radiofrecuencia que permite mayor movilidad a los usuarios al minimizar las conexiones cableadas.*

# Consideraciones de Diseño

## Suposiciones

*Las suposiciones y dependencias del sistema y del proyecto probablemente ya han sido señaladas en los documentos del SPMP y el SRS. Esta sección no debería repetirlas. En cambio debería encargarse del contexto del diseño de las suposiciones señaladas en los otros documentos si es necesario, o sacar a relucir las nuevas suposiciones que solo hacen parte del diseño. [7]*

*Se deben listar supuestos del equipo de desarrollo, la maquina donde va a correr la aplicación (Hardware y Software), y el cliente que va a utilizar la aplicación. En la ilustración 3 se muestran las suposiciones de una aplicación general clasificados en los tres grupos generales.*

Ilustración : Suposiciones generales

## Restricciones

*Las restricciones son aquellas limitaciones externas que impidan que el sistema alcance sus metas. Y al igual que las suposiciones y dependencias, estas ya han sido señaladas en el SPMP y el SRS, por lo tanto, deberían ser repetidas en este documento solo si se les añade algún detalle extra o si algún contexto especifico es añadido. [7]*

*Se deben listar restricciones generales, características del usuario, restricciones del software y restricción de hardware. En la ilustración 4 se muestran las restricciones de una aplicación general.*

Ilustración : Restricciones

## Entorno del Sistema

*Si este no ha sido cubierto en otro documento del proyecto, se debe describir los requerimientos del entorno del sistema, y se debe incluir el entorno del hardware y del software, y cualquier otro entorno si es necesario (por ejemplo, discutir el ambiente de desarrollo si este es diferente). En caso de que esta sección ya haya sido cubierta en algún otro documento es necesario añadir nuevos detalles para entender mejor la perspectiva del diseño, estos detalles adicionales deberían ser cubiertos en esta sección (ver ilustración 5). [7]*

*Se debe describe el entorno a nivel de software y hardware con el que interactuará el sistema.*

* **Software:** La descripción del software con el que contará la aplicación y que será necesario para su adecuado funcionamiento.
* **Hardware:** La descripciones del hardware, que se hacen con base en el entorno en el que se realizarán las pruebas de aceptación del producto.

Ilustración : Entorno del Sistema

## Metodología de Diseño

*Se debe describir brevemente los métodos, procesos, aproximaciones, técnicas, o convenciones usadas para el diseño del software. Las áreas de consideración deberían incluir:*

* *Procesos o políticas impuestas externamente*
* *Balance y compensaciones globales de la ingeniería*
* *Algoritmos o patrones de diseño seleccionados para el sistema*
* *Planes para la trazabilidad, pruebas y mantenimiento de los requerimientos*

*Los elementos señaladas en esta sección deben impactar múltiples áreas del sistema, si los elementos impactan solo un componente especifico, deberían ser señaladas como parte de los detalles de diseño de estos componentes. [7]*

*Una metodología es una colección de métodos para la resolución de una clase de problemas. El desarrollo de metodologías evita los defectos proporcionando técnicas que minimizan la introducción de defectos en los modelos del sistema y en el código. [4]*

Se debería utilizar una ayuda grafica, como por ejemplo una tabla, con el fin de explicar de forma clara las metodologías utilizadas, la importancia y justificación de su uso para la realización del diseño.

## Riesgos

*En los riesgos o áreas volátiles se debe discutir las fuentes más probables de cambio o de riesgo (nuevos requerimientos, tecnología, etc.) que puedan impactar el diseño del sistema. Si es apropiado, se debe describir como el sistema debe ser diseñado para permitir la respuesta oportuna a los cambios o cual es el plan de contingencia para estos cambios. [7]*

*Un riesgo es el área de incertidumbre que puede dar lugar a una desviación en el plan del proyecto (por ejemplo, entrega atrasada, requerimientos no satisfechos, costos mayores o presupuestados, cambios en el diseño, etc.), incluyendo la falla del proyecto. [4]*

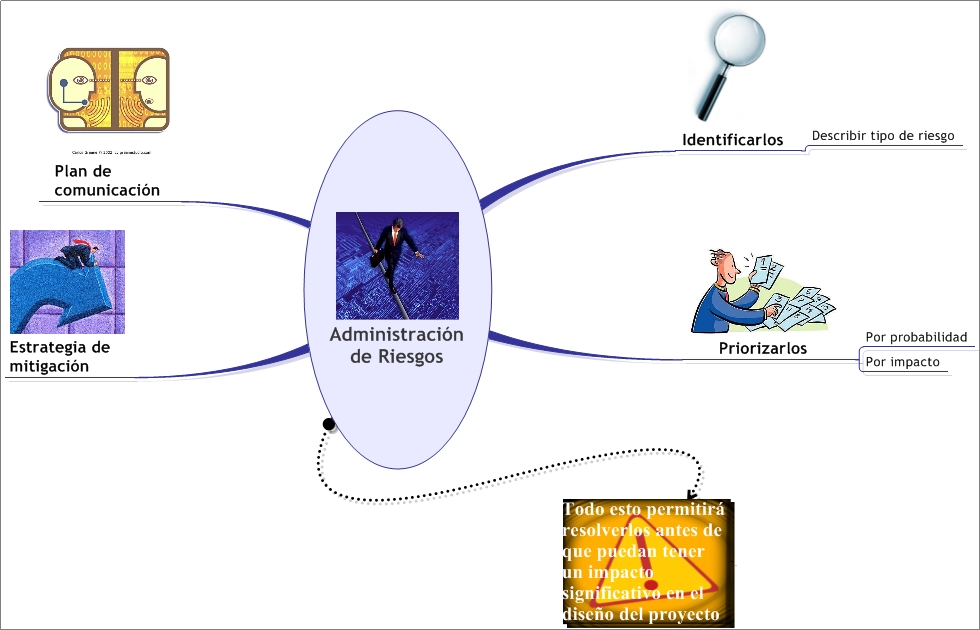
**

Ilustración : Administración de Riesgos

# Arquitectura

En esta sección se especificará la Arquitectura del Sistema que fue escogido para la implementación final de la aplicación. Antes de escoger la arquitectura adecuada es necesario evaluar las ventajas y desventajas de cada una y de esta forma determinar la arquitectura que prestará mayores funcionalidades de desempeño, disponibilidad (y otro atributos de calidad). La muestra un pequeño resumen de las principales características de los estilos arquitectónicos más utilizados.

|  |  |  |
| --- | --- | --- |
| **Estilos** | **Ventajas** | **Desventajas** |
| Tubos y Filtros | * Simplicidad * Soporta ejecución concurrente * Alto nivel de reutilización * Mantenibilidad * Bajo Acoplamiento | * Ejecución por lotes, no permite interacción * Estados de filtros no son compartidos * Topología restringida * Uso de espacio ineficiente * No es aconsejable para sistemas que necesiten interactividad con el usuario |
| Sistemas Basados en Eventos | * Simplicidad * Reutilización * Mantenimiento * Fácil evolución * Manejo Modular permitido | * Poca comunicación o con errores entre el trigger del evento y el componente notificado * Pérdida de control en el comportamiento del sistema * Poca visibilidad, no permite escalabilidad de una forma sencilla |
| N-Tier  Basado en capas | * Descomposición de problemas en varios niveles * Migración de sistemas * Reutilización * Mantenimiento * Abstracción del Sistema | * No todos los sistemas pueden ser estructurados con capas * Difícil determinar la des-composición adecuada * Disminución de rendimiento si la comunicación entre capas de extremos es muy alta |
| Orientado a Objetos | * Cambio en la implementación de objetos puede hacerse sin modificar al cliente * Reutilización * Encapsulamiento de información | * Debe haber conocimiento entre dos objetos que se quieran comunicar * Algunos cambios pueden producir inconsistencia * Efectos colaterales cuando dos objetos usan un tercero |
| Basado en Repositorios | * Resolución de problemas no determinísticos * Útil para sistemas que necesiten acceso de datos compartidos * Fácil de administrar | * Cuellos de botella * Seguridad se vuelve un factor determinante |
| Cliente-Servidor | * Servidor independiente de los clientes (no los conoce) * Aprovecha los sistemas de red * Mantenibilidad * Encapsulamiento * Seguridad | * Intercambio de datos puede ser ineficiente * Administración redundante en cada servidor * Disponibilidad y ubicación de los servidores se vuelve un problema |

Tabla : Ventajas y Desventajas de Estilos Arquitectónicos

Para más información remitirse a [1] [2].

## Apreciación Global

Una vez se ha seleccionado la arquitectura a utilizar para el proyecto, en esta sección se muestran las razones para la escogencia de este modelo.

Ilustración : Apreciación Global

*Básicamente toda aplicación tiene 3 módulos principales 1) Presentación que se encarga de mostrar la información, 2) Lógica el cual realiza el manejo interno de todos los datos y en general del proceso interno de la aplicación y 3) Datos al cual se le delega el almacenamiento, la modificación y la eliminación de la información guardada ya sea en un motor de bases de datos o en un sistema de archivos.*

*Para el proyecto 7 Texas Hold’em del periodo 2007-01 la representación de los módulos [3] (que en ese caso fueron capas, debido a la elección de la arquitectura) puede verse en la .*

Ilustración : Capas Arquitectónicas

## Diagrama de Componentes

*Este diagrama presenta los componentes del sistema, sus interfaces y conectores (ilustración 9), incluyendo la organización y dependencia lógica entre ellos; además, se muestran los subsistemas que conforman la aplicación [4].*

*Una forma de explicar o documentar el diagrama de componentes es haciendo la división por subsistemas y para cada uno explicar los componentes e interfaces asociados. Posibles características que se pueden tener en cuenta en cada descripción son [5]:*

Ilustración : Diagrama de componentes

1. *Composición de los subsistemas: Indica los componentes que se encuentran en el subsistema.*
2. *Propósito de subsistemas y componentes: Describe por qué existen, su importancia y características generales.*
3. *Interfaces disponibles: Esta es sólo para los componentes e indica las interfaces que tiene cada uno disponibles para los demás, las cuales permiten realizar la comunicación entre dichos componentes y utilizar los servicios prestados por cada uno. Esta incluye:*
   1. *Nombre de la interfaz.*
   2. *Componentes que la utilizan.*
   3. *Servicios prestados.*
4. *Dependencias: Esta es sólo para componentes, muestra las interfaces y componentes de los que depende utilizando sus servicios. Esta incluye:*
   1. *Nombre de la interfaz.*
   2. *Componente que la ofrece (el “oferente” de la interfaz).*
   3. *Descripción de la dependencia.*

*Un ejemplo de un diagrama de componentes es el encontrado en [3], de ese trabajo se tomó el subsistema Usuario, , que tenía conexión con el subsistema Aplicación, el cual incluía toda la lógica del juego. Es posible ver los componentes que son las cajas en el diagrama y las interfaces que son los círculos.*

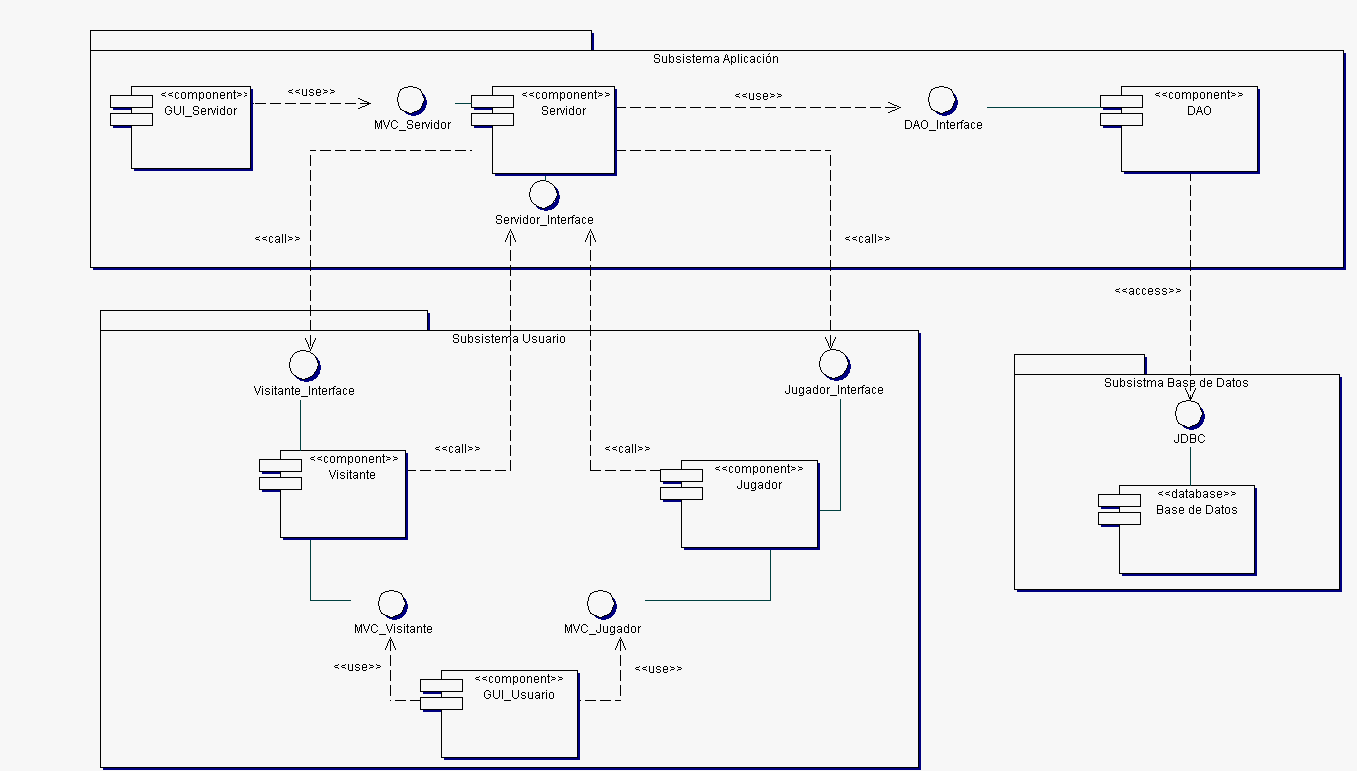


Ilustración : Subsistema Usuario 7 Texas Poker

### Subsistema 1

*En la se muestra un formato que puede ser usado para describir cada uno de los subsistemas de la aplicación final.*

|  |  |
| --- | --- |
| **Propósito del Subsistema** | *Descripción del Propósito* |
| **Composición del Subsistema** | * *Componente 1* * *Componente 2* * *Componente N* |

Tabla : Subsistema 1

### Componente 1

En la se muestra un formato que puede ser usado para describir cada uno de los componentes de la aplicación final.

|  |  |  |  |
| --- | --- | --- | --- |
| **Propósito del**  **Componente** | *Descripción del propósito* | | |
| **Interfaces Disponibles** | *Nombre de la Interfaz* | *Componentes que la utilizan* | *Servicios prestados* |
| *Interfaz 1* | *Componente 2* | *Descripción de la utilidad o función de la interfaz mencionada. ¿Qué servicios presta a los componentes que la utilizan?* |
| **Dependencias** | *Nombre de la Interfaz* | *Componente que la ofrece* | *Descripción de la dependencia* |
| *Interfaz 2* | *Componente 3* | *Descripción de la utilidad o función de la interfaz mencionada. ¿Qué servicios presta a este componente?* |

Tabla : Componente 1

*Esta sección finaliza cuando se hayan documentado todos los subsistemas y sus componentes asociados; para permitir la trazabilidad del sistema (con el uso de este documento) se sugiere utilizar algún tipo de convención, por ejemplo asignar a cada subsistema un color y dejar las tablas con la explicación de los componentes asociados con ese mismo color, de esta forma es más fácil relacionar los subsistemas y sus componentes con su módulo (sección 3.1) correspondiente.*

## Estrategias de Diseño

*Esta sección es un resumen de las diferentes estrategias de diseño que serán utilizadas por el equipo de desarrollo para la implementación de la aplicación, entre estas se pueden incluir [7]:*

Ilustración : Estrategias de Diseño

*Un posible formato que se puede usar para la especificación de las Estrategias de Diseño es el descrito en la*

|  |  |
| --- | --- |
| **Estrategias de diseño** | **Descripción** |
| **Estrategia 1** | * *Razones para la escogencia de la estrategia* * *Utilización de las estrategias dentro del contexto del proyecto* |

Tabla : Estrategias de Diseño

# Diseño de Alto Nivel

## Diagrama de despliegue

*El diagrama de despliegue describe la arquitectura física del sistema en ejecución, es decir, presenta los elementos hardware y software que ejecuta cada uno de ellos.*

*Un Diagrama de Despliegue describe una* ***arquitectura de ejecución****, esto es la configuración de hardware y software que define cómo el sistema estará configurado y además cómo operará (identificación de procesos). El propósito principal de un diagrama de este tipo es presentar una vista estática o “foto” del ambiente de implementación [6].*

*Una forma de explicar o documentar el diagrama de despliegue es haciendo la división por nodos y por conexiones:*

Ilustración : Descripción de Nodos

* Para los Nodos :
  1. Nombre del nodo: Este es el estereotipo que se le asigna al nodo.
  2. Especificación: Hace referencia a las características que debe tener la máquina.
  3. Ubicación: Lugar en donde se encuentra ubicada la máquina.
  4. Componentes: Enumera los componentes software que se ejecutará en el nodo.
  5. Comentarios adicionales: Notas.

Ilustración : Descripción Conexiones

* Para las Conexiones :
  1. Comunicación: Indica el tipo y los nodos que está comunicando dicha conexión.
  2. Protocolos y herramientas: Como su nombre lo indica, describe el tipo de protocolo que se utiliza en la comunicación y algunas herramientas de software necesarias.
  3. Medio: Indica el medio físico por el cual se realiza la conexión.
  4. Comentarios adicionales: Notas.

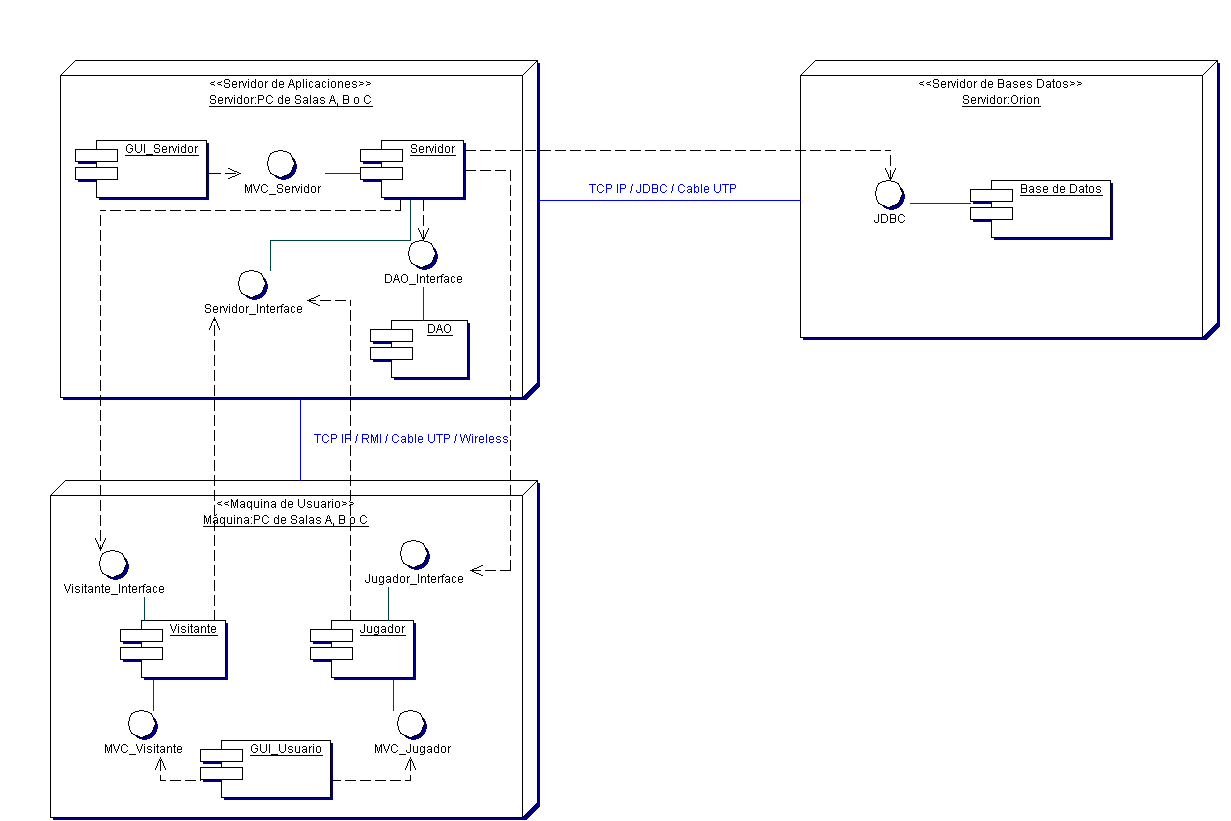
Un ejemplo de un diagrama de despliegue es el encontrado en [3], de ese trabajo se tomó el Nodo Usuario, , que tenía conexión con el Nodo Servidor de Aplicaciones, del cual se observa una sección. Es posible ver los nodos que son los cubos en el diagrama y las conexiones que son las líneas continuas de color azul.

Ilustración : Nodo Usuario 7 Texas Poker

### Nodo 1 *“Nombre del Nodo”*

*En la se muestra un formato que puede ser usado para describir cada uno de los subsistemas de la aplicación final.*

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre del nodo** | *Estereotipo <<Nombre >>* | | |
| **Especificación** | *Remitirse a la sección 2.3 Entorno del Sistema, para la especificación del equipo* | **Ubicación** | *Espacio físico y real del equipo que está siendo descrito* |
| **Componentes** | | **Comentarios adicionales** | |
| * *Componente 1* * *Componente 3* | | * *Subsistemas o componentes albergados* | |

Tabla : Nodo 1

### Conector 1 “Nombre del Nodo”

*En la muestra un formato que puede ser usado para describir cada uno de los subsistemas de la aplicación final.*

|  |  |  |
| --- | --- | --- |
| **Comunicación** | *Nombre de la asociación* | |
| **Protocolos y herramientas** | **Nombre** | **Descripción** |
| *Protocolo 1* | *Descripción del Protocolo 1* |
| *Herramienta 1* | *Descripción de la Herramienta 1* |
| **Medio** | *Elemento de conexión física ( Wireless- Cable UTP)* | |
| **Comentarios adicionales** | - | |

Tabla : Conector 1

## Diagrama de Comportamiento e Interacción

*Los diagramas de comportamiento de UML permiten tener un mejor acercamiento a lo que sucederá cuando la aplicación realice alguna funcionalidad específica, mientras que los diagramas de interacción añaden a esto el flujo de control y de datos [8].*

*Para estos diagramas lo ideal es escoger uno de los Casos de Uso principales del análisis realizado y hacer los diagramas en base a la elección. Una vez se ha seleccionado el caso de uso, también se hace la traza con los requerimientos asociados y se procede a la construcción de los diagramas.*

*Esta sección se divide en Diagramas de Actividad y de Secuencia*

### Diagrama de Actividad

*Una actividad es un paso en la ejecución de un programa, en la cual un trabajo se ejecuta. Esto puede incluir un cálculo, una consulta, manipulación de datos o simplemente verificación de información [6].*

*Los diagramas de actividad generalmente se utilizan para especificar [9]:*

* *Un método*
* *Un caso de uso*
* *Un proceso de negocio (Workflow)*

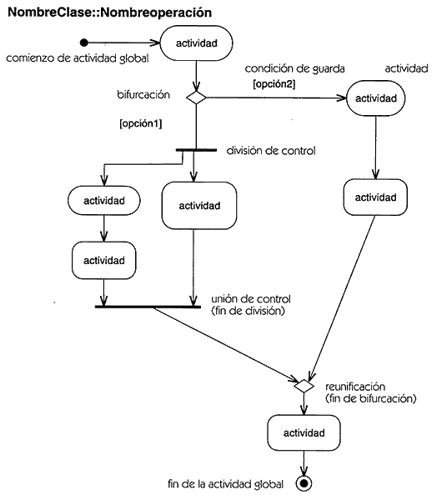


Ilustración : Diagrama de Actividad

### Diagrama de Secuencia

*Los diagramas de secuencia (al igual que los de colaboración) son usados para ilustrar la interacción entre objetos, por lo tanto estos diagramas modelan objetos y paso de mensajes entre objetos. Estos diagramas se hacen visualización basada en el tiempo [6].*

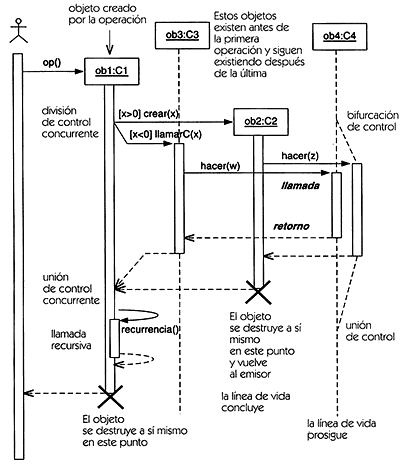


Ilustración : Diagrama de secuencia

# Diseño de Bajo Nivel

*El diseño de bajo nivel describe de forma detallada cómo van a ser implementados los subsistemas, paquetes, módulos y/o componentes que se hayan definido en el diseño de alto nivel [10], para cada uno de estos se deben especificar las clases, interacciones y demás estructuras necesarias para su construcción [11] (ver ilustración 17).*

Ilustración : Diseño de bajo nivel

*Para realizar esta sección se deben llevar a cabo diferentes iteraciones, con las cuales se obtenga un mayor nivel de detalle hasta llegar a uno lo suficientemente claro para una fácil implementación desde el punto de vista del arquitecto. En la ilustración 18 se presentan los diferentes niveles de detalle del sistema de software, indicando cómo después del diseño de alto nivel se empieza con niveles más específicos como lo son la definición de clases con sus métodos y atributos. [12]*

Ilustración : Niveles de diseño

*Para la descripción de bajo nivel se sugiere manejar una estructura jerárquica de los niveles que se hayan definido, en la figura 19 se presenta un ejemplo en el cual se han definido inicialmente subsistemas y estos a su vez se organizan en componentes.*

Ilustración : Ejemplo diseño de bajo nivel

## Subsistema 1

*En esta sección se deben listar todos los componentes, paquetes o módulos que contenga el subsistema 1.*

### Componente 1

*Se debe presentar el diagrama de clases del Componente 1 con su respectiva documentación, sin embargo, para comodidad se sugiere dejar en el presente documento la imagen del diagrama y realizar la documentación en el IDE seleccionado para la implementación con el fin de obtener la caracterización de las clases como un API, por ejemplo, utilizando JavaDoc.*

|  |  |
| --- | --- |
| Descripción de la clase | *Indica el objetivo que cumple la clase y las características propias de esta* |
| Descripción de atributos | *Por cada uno se debe especificar: el tipo de atributo, rango y una breve descripción.* |
| Descripción de métodos | *Por cada uno se debe incluir una breve descripción, la precondición para que el método pueda ser invocado, la poscondición indicando el resultado obtenido después de ser invocado, los parámetros que recibe (de estos debe estar especificado el rango y una breve descripción), el retorno indicando el tipo de dato que se retorna y el rango que posee; por último, se deben incluir las excepciones que va a manejar el método, describiendo en qué casos ocurriría.* |

Tabla : Documentación de clases

*En las ilustraciones 20, 21 y 22 se presenta un ejemplo de documentación de la clase “ConsultasDao” del proyecto 7 Texas Poker.*



Ilustración : Ejemplo Descripción de Clase



Ilustración : Ejemplo Resumen de Métodos



Ilustración : Ejemplo descripción de método

# Diseño de Interfaces de Usuario

## Diseño general de la aplicación

Se debe especificar cada una de las interfaces, indicando el diseño tanto de las entradas como de las salidas que generan información valiosa a los usuarios del sistema.

Para cada una de las interfaces de debe indicar:

Ilustración : Diseño general de la aplicación

Además se debe manejar el siguiente formato para cada una de las entradas y salidas de la interfaz

|  |  |
| --- | --- |
| Convención | *Identificador de la entrada/salida dentro de la interfaz* |
| Nombre | *Nombre de la entrada/salida dado en las documentaciones* |
| Alias (Interfaz) | *Nombre o Label con el que identifican la entrada/salida* |
| Alias (Código) | *Nombre a nivel del código fuente del programa de la entrada/salida* |
| Propósito | *Propósito de funcionalidad de la entrada/salida* |
| Tipo de Entrada | *Por ejemplo: Campo de Texto, Tabla, Botón, etc.* |
| Tipo de Dato | *Por ejemplo: Numérico, Alfanumérico o Alfabético* |
| Longitud mínima | *Longitud mínima asignada a la entrada/salida* |
| Longitud máxima | *Longitud máxima asignada a la entrada/salida* |
| Valor por defecto | *Valor por defecto que se encuentra en la entrada/salida* |
| Rango | *Rango de los valores que puede tener la entrada/salida* |
| Validación | *Como se verifica que la entrada/salida es valida* |
| Secuencia de Entrada | *Orden en la secuencia de entradas/salidas de la interfaz* |
| Comentarios | *Comentarios como por ejemplo: obligatoriedad, medio por el cual entra o sale* |

Tabla : Descripción de entradas y salidas

## Árbol de navegabilidad

En el árbol de navegabilidad de las interfaces gráficas, cada nodo representa el título de dicha interfaz y las flechas los posibles caminos





Ilustración : Ejemplo árbol de navegabilidad

# Anexos

**REFERENCIAS**

[1] Garlan D. et al, An Introduction to Software Architecture, Enero 1994, Carnegie Mellon University

[2] Avila J, Estilos Arquitectónicos, Disponible en http://sophia.javeriana.edu.co/~javila/arquitectura/estilosArquitectonicos.pdf

[3] IronWorks, Descripción Del Diseño De Software 7 Texas Poker, Primer Semestre 2007, Pontificia Universidad Javeriana

[4] Bruegge B, Dutoit AH. Ingeniería de Software orientada a objetos. 1st ed. Trujano G. México: Pearson Educación; 2002.

[5] IEEE (Institute of Electrical and Electronics Engineers), IEEE Recommended Practice for Software Design Descriptions, IEEE-SA Standards Board, Septiembre 1998.

[6] Pender T, UML Bible, Wiley Publishing Inc. Publicado en 2003

[7] Construx Software, DESIGN - CXOne Standard, Construx Software Builder, Inc, Noviembre 2002.

[8] Larman C. UML Y PATRONES. Una introducción al análisis y diseño orientado a objetos y al pro-ceso unificado. 2nd ed. Aragon DF. Madrid: Pearson Educación. S.A.; 2003.

[9] Booch G. et al, El lenguaje Unificado de Modelado, Manual de Referencia. Rational Software Corporation. Publicado en 2000

[10] Construx Software, DESIGN - CXOne Guide, Construx Software Builder, Inc, Noviembre 2002.

[11] Gill C. CSE 432S: Project Low Level Design. Washington University, Departamento de Ciencia y Computación, Febrero 2006.

[12] Mcconnell S. Code Complete, capítulo 5 Desing in Construction. Segunda edición. Disponible en: ww.cc2e.com/File.ashx?cid=336