Presentación

Datos Personales.

Nombre: Ezequiel Garcia M.

Matricula: 2014-0476.

Asignatura.

Asignatura: Construcción de software.

Código: Sof-111.

Sección: 01.

Descripción del Proyecto propuesto.

Título: Resa.

Descripción: Proyecto final de la asignatura construcción de software otorgada por Luis Besewhell Feliz.

Sistema Planteado: Sistema de reservación de salones destinados a eventos.

Profesor.

Nombre: Luis Besewhell Feliz.

Marco Teórico y justificación.

Sistema de reservación de salones de eventos.

Se pretende desarrollar un software que puede ser aplicado como una herramienta útil para la reservación de salones arrendados para eventos. Este documento se estará enmarcando definir la metodología de desarrollo del sistema que permita a los desarrolladores o usuarios seguir obtener algunas especificaciones en cada una de las etapas de del desarrollo del sistema desde los requerimientos iniciales hasta las pruebas finales, que haga que el software sea coherente y además y le dé formalidad al mismo.

A continuación se abordaran los conceptos computacionales tomados en cuenta durante la elaboración del software de este proyecto nombrado (Resa).

Autor: Ezequiel Garcia Mancebo.

* Ingeniería del software

El establecimiento y uso de principios de ingeniería robustos, orientados a obtener económicamente software que sea fiable y funcione eficientemente sobre máquinas reales.

La Ingeniería del Software incluye la aplicación práctica del conocimiento científico en el diseño y construcción de los programas y la documentación requerida para su desarrollo, operación y mantenimiento.

* La ingeniería del Software abarca los siguientes puntos.
* Arquitectura del Software.

La Arquitectura del Software es el diseño de más alto nivel de la estructura de un sistema.

* Construcción de software.

El término Software Construction (Construcción del software) se refiere a la creación de software productivo y significativo a través de los procesos decodificación, verificación, pruebas unitarias, pruebas de integración y depuración de errores.

* Prueba de software.

Demostrar al desarrollador y al cliente que el software alcanza sus requisitos.

* Documentación.

La documentación de sistemas es el conjunto de información que nos dicen que hacen los sistemas como lo hacen y para quien lo hace.

La documentación adecuada y completa, de una aplicación que se desea implantar, mantener y actualizar en forma satisfactoria.

* Arquitectura de software.

Una Arquitectura de Software, también denominada *Arquitectura lógica*, consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco.

La Arquitectura del Software es el diseño de más alto nivel de la estructura de un sistema.

Tipos de arquitectura

**Descomposición Modular**. Donde el software se estructura en grupos funcionales muy acoplados.

**Cliente-servidor.** Donde el software reparte su carga de cómputo en dos partes independientes pero sin reparto claro de funciones.

**Arquitectura de tres niveles**. Especialización de la arquitectura cliente-servidor donde la carga se divide en tres partes (o capas) con un reparto claro de funciones: una capa para la presentación (interfaz de usuario), otra para el cálculo (donde se encuentra modelado el negocio) y otra para el almacenamiento (persistencia). Una capa solamente tiene relación con la siguiente.

* Programación por capas

La programación por capas es una arquitectura cliente-servidor en el que el objetivo primordiales la separación de la lógica de negocios de la lógica de diseño; un ejemplo básico de esto consiste en separar la capa de datos de la capa de presentación al usuario.

* Plataforma

En [informática](https://es.wikipedia.org/wiki/Inform%C3%A1tica), una plataforma es un sistema que sirve como base para hacer funcionar determinados módulos de [hardware](https://es.wikipedia.org/wiki/Hardware) o de [software](https://es.wikipedia.org/wiki/Software) con los que es [compatible](https://es.wikipedia.org/wiki/Compatibilidad_(inform%C3%A1tica)). Dicho sistema está definido por un estándar alrededor del cual se determina una [arquitectura de hardware](https://es.wikipedia.org/w/index.php?title=Arquitectura_de_hardware&action=edit&redlink=1) y una [plataforma de software](https://es.wikipedia.org/wiki/Plataforma_de_software)(incluyendo [entornos de aplicaciones](https://es.wikipedia.org/w/index.php?title=Entorno_de_aplicaciones&action=edit&redlink=1)). Al definir plataformas se establecen los tipos de [arquitectura](https://es.wikipedia.org/wiki/Arquitectura_(inform%C3%A1tica)), [sistema operativo](https://es.wikipedia.org/wiki/Sistema_operativo), [lenguaje de programación](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n) o [interfaz de usuario](https://es.wikipedia.org/wiki/Interfaz_de_usuario) compatibles.

* Ambiente \*-\*

Un entorno es un espacio o escenario informático en donde operan determinados comandos, funciones o características comunes.

* Infraestructura \*-\*

Es el conjunto de [hardware](http://es.wikipedia.org/wiki/Hardware) y [software](http://es.wikipedia.org/wiki/Software) sobre el que se asientan los diferentes sistemas.

* UI o (user interface)

La interfaz de usuario es el medio con que el usuario puede comunicarse o interactuar con una máquina, equipo, [computadora](https://es.wikipedia.org/wiki/Computadora) o dispositivo, y comprende todos los puntos de contacto entre el [usuario](https://es.wikipedia.org/wiki/Usuario) y el equipo.

* GUI (user graphics interface)

La interfaz gráfica de usuario, conocida también como GUI (del [inglés](https://es.wikipedia.org/wiki/Idioma_ingl%C3%A9s) graphical user interface), es un [programa informático](https://es.wikipedia.org/wiki/Programa_inform%C3%A1tico) que actúa de [interfaz de usuario](https://es.wikipedia.org/wiki/Interfaz_de_usuario), utilizando un conjunto de imágenes y [objetos gráficos](https://es.wikipedia.org/wiki/Icono) para representar la información y acciones disponibles en la interfaz.

* GDI (Graphics device interface)

Graphics Device Interface (GDI) es uno de los tres componentes o subsistemas de la [interfaz de usuario](https://es.wikipedia.org/wiki/Interfaz_de_usuario) de [Microsoft Windows](https://es.wikipedia.org/wiki/Microsoft_Windows). Trabaja junto con el [núcleo](https://es.wikipedia.org/wiki/N%C3%BAcleo_(inform%C3%A1tica)) y la [API de Windows](https://es.wikipedia.org/wiki/API_de_Windows).

Esta [interfaz de programación de aplicaciones](https://es.wikipedia.org/wiki/Interfaz_de_programaci%C3%B3n_de_aplicaciones) se encarga del control gráfico de los dispositivos de salida, como los [monitores](https://es.wikipedia.org/wiki/Monitor_de_computadora) o las [impresoras](https://es.wikipedia.org/wiki/Impresora).

* Temas o themes

Un tema (skin, piel en [inglés](https://es.wikipedia.org/wiki/Idioma_ingl%C3%A9s)), también llamado theme, tema o tapiz, es una serie de elementos gráficos que, al aplicarse sobre un determinado [software](https://es.wikipedia.org/wiki/Software), modifican su apariencia externa.

* UX (User xperience)

La experiencia de usuario es el conjunto de factores y elementos relativos a la interacción del [usuario](https://es.wikipedia.org/wiki/Usuario_(inform%C3%A1tica)), con un entorno o dispositivo concretos, cuyo resultado es la generación de una percepción positiva o negativa de dicho servicio, producto o dispositivo.

* Lenguajes de programación

Un lenguaje de programación no es más que un sistema estructurado y diseñado principalmente para que las máquinas y computadoras se entiendan entre sí y con nosotros, los humanos. Contiene un conjunto de acciones consecutivas que el ordenador debe ejecutar.

* Generaciones de los lenguajes de programación.

Los lenguajes de programación se dividen en 2 categorías fundamentales:

Bajo nivel: Son dependientes de la máquina, están diseñados para ejecutarse en una determinada computadora. A esta categoría pertenecen las 2 primeras generaciones.

Alto Nivel: Son independientes de la máquina y se pueden utilizar en una variedad de computadoras. Pertenecen a esta categoría la tercera y la cuarta generación. Los lenguajes de más alto nivel no ofrecen necesariamente mayores capacidades de programación, pero si ofrecen una *interacción programador/computadora más avanzada*. Cuanto más alto es el nivel del lenguaje, más sencillo es comprenderlo y utilizarlo.

Cada generación de lenguajes es más fácil de usar y más parecida a un lenguaje natural que su predecesores. Los lenguajes posteriores a la cuarta generación se conocen como *lenguajes de muy alto nivel*. Son lenguajes de muy alto nivel los generadores de aplicaciones y los naturales.

En cada nuevo nivel se requieren menos instrucciones para indicar a la computadora que efectúe una tarea particular. Pero los lenguajes de alto nivel son sólo una ayuda para el programador. Un mayor nivel significa que son necesarios menos comandos, debido a que cada comando o mandato de alto nivel reemplaza muchas instrucciones de nivel inferior.

1. Primera - Lenguaje de máquina: Empieza en los años 1940-1950. Consistía en sucesiones de dígitos binarios. Todas las instrucciones y mandatos se escribían valiéndose de cadenas de estos dígitos. Aún en la actualidad, es el único lenguaje interno que entiende la computadora; los programas se escriben en lenguajes de mayor nivel y se traducen a lenguaje de máquina.

2. Segunda - Lenguajes ensambladores: Fines de los ´50. Se diferencian de los lenguajes de máquina en que en lugar de usar códigos binarios, las instrucciones se representan con símbolos fáciles de reconocer, conocidos como *mnemotécnicos*. Aún se utilizan estos lenguajes cuando interesa un nivel máximo de eficiencia en la ejecución o cuando se requieren manipulaciones intrincadas. Al igual que los lenguajes de la máquina, los lenguajes ensambladores son únicos para una computadora particular. Esta dependencia de la computadora los hace ser lenguajes de bajo nivel.

3. Tercera: años 60. Los lenguajes de esta generación se dividen en tres categorías, según se orienten a:

Procedimientos: Requieren que la codificación de las instrucciones se haga en la secuencia en que se deben ejecutar para solucionar el problema. A su vez se clasifican en científicos (p.ej.: FORTRAN), empresariales (v.g.: COBOL), y de uso general o múltiple (p.ej.: BASIC). Todos estos lenguajes permiten señalar *cómo* se debe efectuar una tarea a un nivel mayor que en los lenguajes ensambladores. Hacen énfasis los procedimientos o las matemáticas implícitas, es decir en *lo* que se hace (la acción).

Problemas: Están diseñados para resolver un conjunto particular de problemas y no requieren el detalle de la programación que los lenguajes orientados a procedimientos. Hacen hincapié en la entrada y la salida deseadas.

4. Cuarta: su característica distintiva es el énfasis en especificar *qué* es lo que se debe hacer, en vez de como ejecutar una tarea. Las especificaciones de los programas se desarrollan a un más alto nivel que en los lenguajes de la generación anterior. La característica distintiva es ajena a los procedimientos, el programador no tiene que especificar cada pasó para terminar una tarea o procesamiento.

* Lenguaje C#

C# (C Sharp): es un lenguaje de programación orientada a objetos, creado por danés Anders Hejlsberg y estandarizado por Microsoft como parte de la plataforma .NET, sus sintaxis es derivada de los leguajes de programación C/C++, se utiliza mayormente para el desarrollo de sistemas sobre la plataforma de Windows, aunque existe compiladores como Mono el cual escribes código c# pata distintas plataformas como Linux, Android, IOS, Mac X OS, Windows Phone.

Versiones de c#:

* C# 1.0: fue el primer idioma que el desarrollador a adoptado para construir .NET 2002.
* C# 2.0: tiene nuevas características introducidas en esta edición, que ayudaron a los desarrolladores al código de sus aplicaciones de una manera más genérica. Entre sus características están Métodos genéricos, Métodos anónimos, tipos de datos, Nullables, Clase Parcial, Covarianza y Contra-Varianza.(2005)
* C# 3.0: estas son las características de esta versión: Expresiones Lambda, Métodos de extensión, Árbol de extensión, Tipos anónimos, LINQ, Tipo implícito (var).(2007)
* C# 4.0: estas son las características de esta versión: Binding retrasado, Argumentos con nombre, Parámetros opcionales, Mas soporte COM.(2010)
* C# 5.0: es la más utilizada por los desarrolladores a pesar de que ya existe una versión 6 del lenguaje sus principal característica es: Programación Asincrónica.(2012)
* C# 6.0: esta nueva versión del lenguaje c# presentada el año 2014 nos trae muchas novedades para los desarrolladores las cuales son:
  + Auto-Propiedades solo con los getters
  + Inicialización de auto-properties
  + Uso de clases estáticas
  + Interpolación de la cadenas
  + Métodos que solo tienen una expresión
  + Inicialización de índices
  + Operadores condicionales que comprueban los Null
  + Operador nameof.

Todas estas características nos llevaron a seleccionar este lenguaje como lenguaje de desarrollo del sistema en la versión 5.0.

* Threads o hilos.

Un lenguaje de programación no es más que un sistema estructurado y diseñado principalmente para que las máquinas y computadoras se entiendan entre sí y con nosotros, los humanos. Contiene un conjunto de acciones consecutivas que el ordenador debe ejecutar.

* Base de datos

Una base de datos o banco de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso.

### Las bases de datos pueden clasificarse de varias maneras, de acuerdo al contexto que se esté manejando, la utilidad de las mismas o las necesidades que satisfagan.

### Según la variabilidad de la base de datos

#### Bases de datos estáticas

### Son bases de datos únicamente de lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones, tomar decisiones y realizar análisis de datos para [inteligencia empresarial](https://es.wikipedia.org/wiki/Inteligencia_empresarial).

#### Bases de datos dinámicas

### Las bases de datos pueden clasificarse de varias maneras, de acuerdo al contexto que se esté manejando, la utilidad de las mismas o las necesidades que satisfagan.

### Según el contenido

#### Bases de datos bibliográficas

#### Las bases de datos pueden clasificarse de varias maneras, de acuerdo al contexto que se esté manejando, la utilidad de las mismas o las necesidades que satisfagan.

#### Bases de datos o "bibliotecas" de información química o biológica

#### Son bases de datos que almacenan diferentes tipos de información proveniente de la [química](https://es.wikipedia.org/wiki/Qu%C3%ADmica), las [ciencias de la vida](https://es.wikipedia.org/wiki/Ciencias_de_la_vida) o [médicas](https://es.wikipedia.org/wiki/Medicina).

## Modelos de bases de datos

## Además de la clasificación por la función de las bases de datos, éstas también se pueden clasificar de acuerdo a su modelo de administración de datos. Un modelo de datos es básicamente una "descripción" de algo conocido como contenedor de datos (algo en donde se guarda la información), así como de los métodos para almacenar y recuperar información de esos contenedores.

## Bases de datos jerárquica

En este modelo los datos se organizan en forma de árbol invertido (algunos dicen raíz), en donde un nodo padre de información puede tener varios hijos. El nodo que no tiene padres es llamado raíz, y a los nodos que no tienen hijos se los conoce como hojas.

Las bases de datos jerárquicas son especialmente útiles en el caso de aplicaciones que manejan un gran volumen de información y datos muy compartidos permitiendo crear estructuras estables y de gran rendimiento.

### Base de datos de red

Éste es un modelo ligeramente distinto del jerárquico, su diferencia fundamental es la modificación del concepto de nodo: se permite que un mismo nodo tenga varios padres (posibilidad no permitida en el modelo jerárquico).

Fue una gran mejora con respecto al modelo jerárquico, ya que ofrecía una solución eficiente al problema de redundancia de datos, pero, aun así, la dificultad que significa administrar la información en una base de datos de red ha significado que sea un modelo utilizado en su mayoría por programadores más que por usuarios finales.

### Bases de datos transaccionales

Son bases de datos cuyo único fin es el envío y recepción de datos a grandes velocidades, estas bases son muy poco comunes y están dirigidas por lo general al entorno de análisis de calidad, datos de producción e industrial, es importante entender que su fin único es recolectar y recuperar los datos a la mayor velocidad posible, por lo tanto la redundancia y duplicación de información no es un problema como con las demás bases de datos, por lo general para poderlas aprovechar al máximo permiten algún tipo de conectividad a bases de datos relacionales.

### Bases de datos relacionales

Éste es el modelo utilizado en la actualidad para representar problemas reales y administrar datos dinámicamente. Tras ser postulados sus fundamentos en [1970](https://es.wikipedia.org/wiki/1970) por [Edgar Frank Codd](https://es.wikipedia.org/wiki/Edgar_Frank_Codd), de los laboratorios [IBM](https://es.wikipedia.org/wiki/IBM) en [San José (California)](https://es.wikipedia.org/wiki/San_Jos%C3%A9_(California)), no tardó en consolidarse como un nuevo paradigma en los modelos de base de datos. Su idea fundamental es el uso de "relaciones". Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados "[tuplas](https://es.wikipedia.org/wiki/Tupla)". Pese a que ésta es la teoría de las bases de datos relacionales creadas por Codd, la mayoría de las veces se conceptualiza de una manera más fácil de imaginar. Esto es pensando en cada relación como si fuese una tabla que está compuesta por [registros](https://es.wikipedia.org/wiki/Registro_(base_de_datos)) (las filas de una tabla), que representarían las tuplas, y [campos](https://es.wikipedia.org/wiki/Campo_(base_de_datos)) (las columnas de una tabla).

### Bases de datos multidimensionales

Son bases de datos ideadas para desarrollar aplicaciones muy concretas, como creación de [Cubos OLAP](https://es.wikipedia.org/wiki/Cubo_OLAP). Básicamente no se diferencian demasiado de las bases de datos relacionales (una tabla en una base de datos relacional podría serlo también en una base de datos multidimensional), la diferencia está más bien a nivel conceptual; en las bases de datos multidimensionales los campos o atributos de una tabla pueden ser de dos tipos, o bien representan dimensiones de la tabla, o bien representan métricas que se desean aprender.

### Bases de datos orientadas a objetos

Este modelo, bastante reciente, y propio de los [modelos informáticos orientados a objetos](https://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos), trata de almacenar en la base de datos los objetos completos (estado y comportamiento).

Una base de datos orientada a objetos es una base de datos que incorpora todos los conceptos importantes del paradigma de objetos.

### Bases de datos documentales

Permiten la indexación a texto completo, y en líneas generales realizar búsquedas más potentes, sirven para almacenar grandes volúmenes de información de antecedentes históricos. Tesaurus es un sistema de índices optimizado para este tipo de bases de datos.

### Bases de datos deductivas

Un sistema de base de datos deductiva, es un sistema de base de datos pero con la diferencia de que permite hacer deducciones a través de inferencias. Se basa principalmente en reglas y hechos que son almacenados en la base de datos. Las bases de datos deductivas son también llamadas bases de datos lógicas, a raíz de que se basa en lógica matemática. Este tipo de base de datos surge debido a las limitaciones de la Base de Datos Relacional de responder a consultas recursivas y de deducir relaciones indirectas de los datos almacenados en la base de datos.

* SQL

En la actualidad SQL es el estándar de facto de la inmensa mayoría de los SGBD comerciales. Y, aunque la diversidad de añadidos particulares que incluyen las distintas implementaciones comerciales del lenguaje es amplia, el soporte al estándar SQL-92 es general y muy amplio.

**Inyección SQL**

Inyección SQL es un método de infiltración de código intruso que se vale de una [vulnerabilidad informática](https://es.wikipedia.org/wiki/Error_de_software) presente en una aplicación en el nivel de validación de las entradas para realizar operaciones sobre una [base de datos](https://es.wikipedia.org/wiki/Base_de_datos).

El origen de la vulnerabilidad radica en el incorrecto chequeo o filtrado de las variables utilizadas en un programa que contiene, o bien genera, código [SQL](https://es.wikipedia.org/wiki/SQL). Es, de hecho, un error de una clase más general de vulnerabilidades que puede ocurrir en cualquier [lenguaje de programación](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n) o [script](https://es.wikipedia.org/wiki/Lenguaje_interpretado) que esté embebido dentro de otro.

**Proveedores de datos .NET**

Los proveedores de datos .NET Framework sirven para conectarse a una base de datos, ejecutar comandos y recuperar resultados. Esos resultados se procesan directamente, se colocan en un [DataSet](https://msdn.microsoft.com/es-es/library/system.data.dataset(v=vs.110).aspx) con el fin de que el usuario pueda verlos cuando los necesite, se combinan con datos de varios orígenes o se utilizan de forma remota entre niveles. Los proveedores de datos .NET Framework son ligeros, de manera que crean un nivel mínimo entre el origen de datos y el código, con lo que aumenta el rendimiento sin sacrificar funcionalidad.

En la tabla siguiente se enumeran los proveedores de datos que se incluyen en .NET Framework.

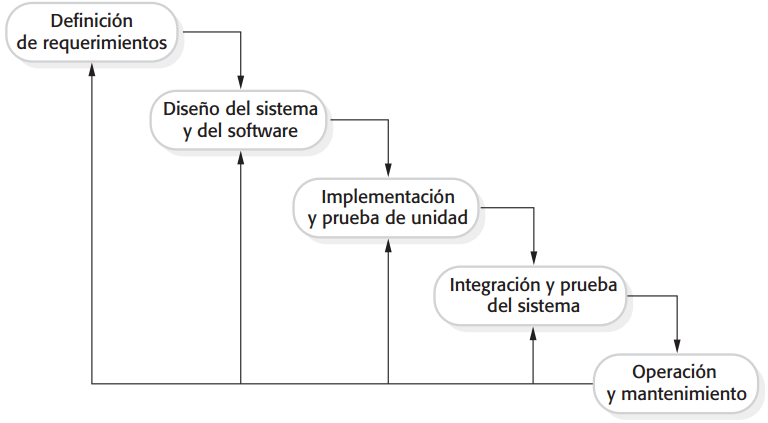
|  |  |
| --- | --- |
| Proveedor de datos .NET Framework | Descripción |
| .NET Framework Proveedor de datos para SQL Server | Proporciona acceso a datos para Microsoft SQL Server. Utiliza el espacio de nombres [System.Data.SqlClient](https://msdn.microsoft.com/es-es/library/system.data.sqlclient(v=vs.110).aspx). |
| Proveedor de datos.NET Framework para OLE DB | Para orígenes de datos que se exponen mediante OLE DB. Utiliza el espacio de nombres [System.Data.OleDb](https://msdn.microsoft.com/es-es/library/system.data.oledb(v=vs.110).aspx). |
| .NET Framework para OBDC | Para orígenes de datos que se exponen mediante ODBC. Utiliza el espacio de nombres [System.Data.Odbc](https://msdn.microsoft.com/es-es/library/system.data.odbc(v=vs.110).aspx). |
| .NET Framework para Oracle | Para orígenes de datos de Oracle. El proveedor de datos .NET Framework para Oracle es compatible con la versión 8.1.7 y posteriores del software de cliente de Oracle y utiliza el espacio de nombres [System.Data.OracleClient](https://msdn.microsoft.com/es-es/library/system.data.oracleclient(v=vs.110).aspx). |
| Proveedor para EntityClient | Proporciona acceso a datos para las aplicaciones de Entity Data Model (EDM). Utiliza el espacio de nombres [System.Data.EntityClient](https://msdn.microsoft.com/es-es/library/system.data.entityclient(v=vs.110).aspx). |
| El proveedor de datos .NET Framework para SQL Server Compact 4.0. | Proporciona acceso de datos para Microsoft SQL Server Compact 4.0. Usa el espacio de nombres [System.Data.SqlServerCe](https://msdn.microsoft.com/en-us/library/system.data.sqlserverce.aspx). |

* Modelos de desarrollo de un software

Un modelo de proceso de software es una representación simplificada de este proceso. Cada modelo del proceso representa a otro desde una particular perspectiva y, por lo tanto, ofrece sólo información parcial acerca de dicho proceso estos son algunos modelo o metodologías de desarrollo de software.

* Modelo en cascada:

Este modelo se conoce como “modelo en cascada” o ciclo de vida del software es un ejemplo de un proceso dirigido por un plan.



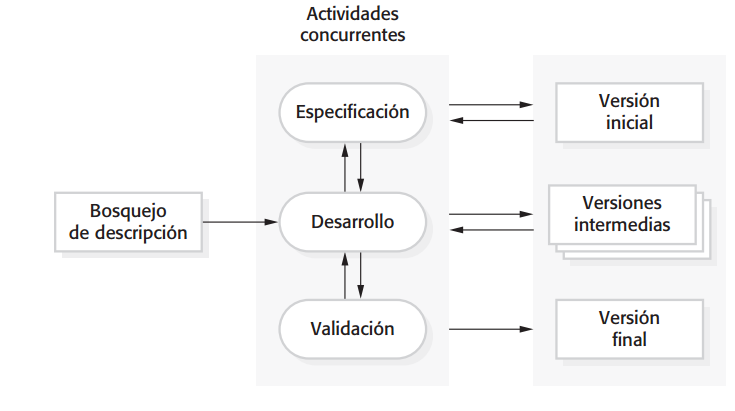
* Modelo ágiles:

El modelo ágil fue desarrollado a razón de que la empresa es tan en vueltas en un mercado Global y el enfoque que las tendencias pueden cambiar rápidamente y que un software diseñado con la metodología dirigida por un plan Podría quedar obsoleto antes de su entrega y como el enfoque de las empresa se mantiene en un entorno cambiante y sus requerimiento varia drástica mente es desarrollado una nueva tendencia al desarrollo.

Hay muchas tendencias del modelo agiles pero estas son unas de las más comunes.

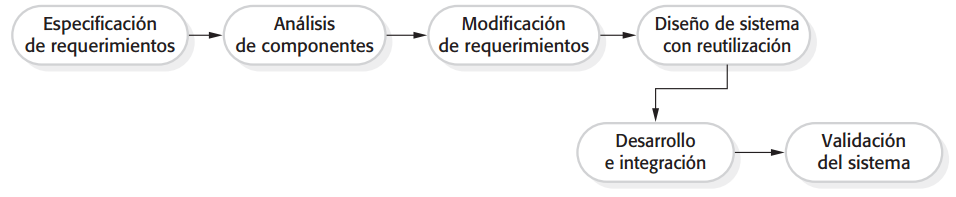
* Los procesos de especificación, diseño e implementación están entrelazados.
* El sistema se desarrolla en diferentes versiones.
* Las interfaces de usuario del sistema se desarrollan usando con frecuencia un
* sistema de elaboración interactivo.
* Modelo de Desarrollo incremental:

El desarrollo incremental se basa en la idea de diseñar una implementación inicial, exponer ésta al comentario del usuario, y luego desarrollarla en sus diversas versiones hasta producir un sistema adecuado.



* Modelo orientada a la reutilización:

En la mayoría de los proyectos de software hay cierta reutilización de software. Cuando las personas que trabajan en el proyecto conocen diseños o códigos que son similares a lo que se requiere. Es común utilizar herramientas del sistema como editores de textos calculadora y herramienta Prediseñadas.



* Modelo de Desarrollo Rápido:

El Desarrollo Rápido de Aplicaciones o RAD (Rapid Application Development): es aquel método que contempla un desarrollo de modo iterativo así como la realización de prototipos, su esencia se concentra en la usabilidad y utilidad así como la rapidez de ejecución.

ETAPAS DEL DESARROLLO RÁPIDO DE APLICACIONES

Modelado de la gestión: Dentro de esta etapa, el objetivo es la solución de las preguntas; por ejemplo:

¿Qué información conduce el proceso de gestión?,

¿Qué información se genera?, ¿A donde va a parar esa información? Etc.

Modelado de datos: Contempla la definición de las características de los objetos, así como la constitución de los objetos y sus vínculos entre ellos.

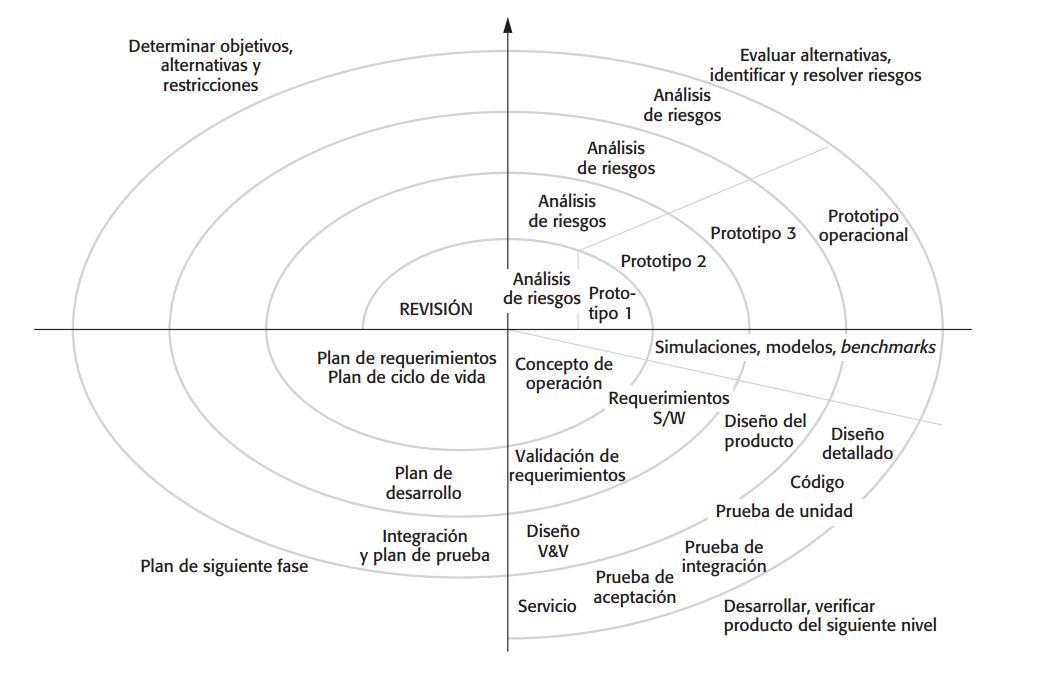
Modelado de proceso: Describe las metodologías que manipulan los objetos así como la comunicación entre ellos.

Generación de aplicaciones: Permite la utilización de recursos que ya existen o crear componentes reutilizables.

Pruebas de entrega: Prueba todos los componentes nuevos.

* Modelo en espiral de Boehm:

Ciclo de Vida en Espiral tiene en cuenta fuertemente el riesgo que aparece a la hora de desarrollar software. Para ello, se comienza mirando las posibles alternativas de desarrollo, se opta por la de riesgo más asumible y se hace un ciclo de la espiral.



* Justificación

En muchos centros públicos e instituciones privadas hoy en día tienen que cargar con el problema de no poder gestionar los salones de los cuales disponen de una manera efectiva y automatizada, con el fin de poder manipular los itinerarios e manejar los eventos programados en el mismo.

El desarrollo de este proyecto permitirá a esta comunidad empresarial:

* Automatización de las solicitudes a eventos.
* Control de eventos en los salones manipulación y candelarizacion.
* Control de reportes.
* En relación con los solicitantes tendrán la oportunidad de ser notificados en cuanto se apruebe su evento.

Con la implementación de este proyecto se busca beneficiar directamente a

la comunidad educativa la cual en muchos de los establecimientos no tienen un control de los salones, se espera que la puesta en práctica del mismo brinda una solución concreta del problema propuesto como objetivo de solución.

Definición de Problemas y objetivos.

Definición de problema

Actualmente los salones de eventos del ITSC (Instituto Técnico Superior Comunitario) no cuentan con un software que gestione su reservación, En la cual a los salones se le asignen los eventos programados por las diferentes facultades de la institución o entidades que quieran programar un evento.

Propuesta de solución

Ante el problema descrito Se optó por elaborar una herramienta de Reservación de los salones nombrada Resa (Re servador de salones) en el ambiente WinForm Con almacenamiento y distribución de datos Cliente servidor con el SGDB (Sistema manejado de bases de datos) SQL Server implementado las fases de desarrollo (Análisis, Arquitectura, Construcción, prueba, implementación) utilizando herramientas como UML (Universal modeling languaje).

Objetivo general

Desarrollar una aplicación interactiva que permita la reservación de salones y consulta de las agendas de dichos salones también la creación de nuevos salones.

Objetivos específicos

* Que se presenta una interfaz con buena usabilidad.
* Autentificación de usuario.
* Que se puedan generar y acoplar nuevos salones.
* Generación de reportes entre.
* Generar solicitudes y a la vez las mismas aprobarse.

Alcances

Con el desarrollo del Sistema de reservación se logrará elaborar un producto que automatizara un problema existente en muchos establecimientos públicos y privados en el país en los cuales se carece de la automatización de alquiler y reservación de salones.

* El sistema creara y gestionara los eventos y salones de diferentes tipos de instalaciones administrando estos procesos.
* El sistema generara reportes de los salones y eventos programados.
* El sistema llevara un inventario de las solicitudes para la reservación de un salón.

Plataforma

A continuación se describirá las características a tener de la plataforma donde será implantado el sistema desarrollado con el proyecto:

* Procesador Intel Core2duo o AMD Athlon X2 a partir de 2.0 GHz.
* Disco duro HDD 80GB o Disco de estado sólido SSD 128GB
* Memoria RAM 2GB
* Tarjeta de video integrada 128MB o dedicada 512MB
* Tarjeta de Red

Arquitectura

**Arquitectura Cliente-Servidor:** es un modelo en la cual la carga de divide en dos, el cliente, el cual es quien contiene las interfaces gráficas y hace las peticiones al servidor. El Servidor sirve como contenedor de datos y tomas las peticiones del cliente mostrado una respuesta inmediata de manera del que el cliente pueda visualizarla.

Elegido este tipo de arquitectura por la versatilidad donde el software reparte su carga de cómputo en dos partes independientes pero sin repast claro de funciones la parte del cliente y la del servidor.

Ambiente

La aplicación tendrá un ambiente local o de escritorio

Los formularios Windows Forms son la nueva plataforma de desarrollo de aplicaciones para Microsoft Windows, basados en .NET Framework. Este marco de trabajo proporciona un conjunto de clases claro, orientado a objetos y ampliable, que permite desarrollar complejas aplicaciones para Windows.

La aplicación será desarrollada en un ambiente desktop / WinForms utilizando las herramientas WinForms del UX Devexpress con el cual los WinForms heredaran muchas de las características de este UX (User experience)

Formularios de Windows, por las características del software, es más conveniente trabajar con un ambiente de formularios de Windows, de esta manera la forma de trabajar se vuelve más factible para el desarrollador, y el por ende si el desarrollador está cómodo y en un ambiente favorable el tiempo de construcción del software es menor y, la entrega es más rápida.

Lenguaje

EL lenguaje utilizado para la construcción del sistema es el ya mencionado en el marco teórico del cual se explicaron sus versiones es el robusto C# elegido por variedad y seguridad que brinda será utilizado para desarrollar el sistema en el framework .NET 4.5 en su versión 6.

Base de Datos

Con los fines de desarrollo del sistema se determinó elegir el modelo de bases de datos relacionales por su dinamismo, fácil representación de un problema y el dinamismo de la administración de los datos.

Otras de las características por la que se conllevo a la elección son:

* La contemplación de relaciones entre las entidades como abstracción de la solución.
* La facilidad de relacionar entidades unas con otras de forma lógica.

El proveedor de datos .NET para SQL server es el proveedor utilizado para la construcción del sistema.

Modelo de Desarrollo

EL modelo a utilizar con fines de desarrollo del proyecto es el modelo incremental ya previamente definido en el marco teórico, algunas de las razones por lo cual selecciono este modelo es por la versatilidad y bajo costo de modificaciones en los requerimientos es muy flexible desarrollar un sistema en este modelo y reduce de manera cuantitativa el costo y la fecha de entrega del mismo llevando el siguiente un punto una conexión directa con el cliente o usuario final.

Patrón de Diseño de Desarrollo

“Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software.”

El patrón de diseño utilizado para el desarrollo del Proyecto es el patrón en Ncapas Basándose en el estilo arquitectural en capas se basa en una distribución jerárquica de los roles y las responsabilidades para proporcionar una división efectiva de los problemas a resolver. Los roles indican el tipo y la forma de la interacción con otras capas y las responsabilidades la funcionalidad que implementan.

Nomenclatura de Programación

Programación

|  |  |  |
| --- | --- | --- |
| Tipo | Notación + Contenido | Ejemplo |
| NameSpaces | Pascal Case, Sin Guión Bajo | Negocio. Usuario |
| Clases (Classes) | Pascal, Sin Guion Bajo | Usuarios |
| Funciones y Subs | Pascal, Sin Guión Bajo | ConvertirContenido (…) |
| Propiedades Publicas | Pascal, Sin Guión Bajo | RecordID |
| Propiedades Privadas | Camel [prefijo \_ opcional] | Salonid, \_Salonid |
| Parámetros | Camel, sin abreviaciones | ref int SalonID |
| Controles en Winform | Camel, con prefijo [ejem: TB] | TBUserId |

Bases de datos

|  |  |  |
| --- | --- | --- |
| Tipo | Notación + Contenido | Ejemplo |
| Nomenclatura de Tablas | Pascal Case (excepto SQL) | Colonias, Módulos |
| Campos | Camel Case, Sin Guión Bajo | idRegistro, diaRegistro |
| Funciones | Pascal Case, Sin Guión Bajo | Letra, DiasVencidos |
| Stored Procedures | Camel, Sin Guión Bajo [proc]+[tabla]+[funcion] | Ej 1: procClienteInsert Ej 2: saldoVencidoSelect \* prefijo proc es opcional\* |
| Parámetros en SP | Camel, Sin Guión Bajo | @idCliente |
| Vistas (Views) | Camel, prefijo “vw” | Clientes |

Herramientas, interfaces e IDE’s

Para la desarrollo de esta herramienta se utilizara el lenguaje orientado a objetos C# en el entorno de desarrollo de visual Studio 2015, SQL Server Managment Como SMDB(Sistema Manejador de Bases de Datos) como sistema de manejador de bases de datos relacionales y otros frameworks.

Otras Herramientas:

* Devexpress:

Es una empresa de desarrollo de software fundada en 1998 con sede en Glendale, California. DevExpress inicialmente comenzó a producir controles de interfaz de usuario para Borland Delphi / C ++ Builder y controles ActiveX de Microsoft Visual Studio. En la actualidad, DevExpress tiene productos dirigidos a los desarrolladores que utilizan Delphi / C ++ Builder, Visual Studio y tecnologías HTML5 / JavaScript. DevExpress es un Microsoft Gold Certified Partner con la competencia "Desarrollo de aplicaciones".

* StarUML:

Es una herramienta UML por MKLab. El software se licencia de una versión modificada de GNU GPL hasta 2014, cuando una versión reescrita 2.0.0 fue liberado para pruebas beta bajo una licencia propietaria. Después de ser abandonado por algún tiempo, el proyecto tuvo un renacimiento para pasar de Delphi para Java / Eclipse y luego se detuvo de nuevo.

* Microsoft Visio:

Microsoft Visio es un [software](https://es.wikipedia.org/wiki/Software) de dibujo vectorial para [Microsoft Windows](https://es.wikipedia.org/wiki/Microsoft_Windows).

Herramienta desarrollada por Microsoft en la cual se generan multitudes de diagramas y cronogramas de diferentes estándar del mercado.

* Microsoft Word :

 Es una [aplicación informática](https://es.wikipedia.org/wiki/Aplicaci%C3%B3n_inform%C3%A1tica) orientada al [procesamiento de textos](https://es.wikipedia.org/wiki/Procesador_de_texto). Fue creado por la empresa [Microsoft](https://es.wikipedia.org/wiki/Microsoft), y viene integrado en el [paquete ofimático](https://es.wikipedia.org/wiki/Paquete_de_oficina) denominado [Microsoft Office](https://es.wikipedia.org/wiki/Microsoft_Office).

REQUERIMIENTOS del SOFTWARE

Sistema de reservación de salones de eventos



Prefacio

|  |  |
| --- | --- |
|  |  |
|  | Este documento describe los requerimientos de software del sistema de reservación de salones de eventos Resa (Reservador de Salones), cuyo objetivo principal es Desarrollar una aplicación que gestione dicha reservación de un conglomerado de salones. |
| Alcance | Este documento de requerimientos de software es la base del desarrollo del sistema de reservación de salones de eventos. Describe los siguientes tópicos:   * Requerimientos de usuario. * Requerimientos de usuario. * Requerimientos no funcionales   Este documento no describe las fases de notaciones graficas del Sistema Dichas notaciones se enmarcaran en otros documentos. |

Historia del Documento

|  |  |  |  |
| --- | --- | --- | --- |
| Fecha | Versión | Comentarios | Autores |
| 25/1/2015 | 0.1 | Versión inicial | Ezequiel G. |
|  |  |  |  |
|  |  |  |  |

* INTRODUCCION

## Entorno

En este documento se desglosara el levantamiento de requerimientos de un sistema de reservación de salones de eventos (Resa) cuyo objetivo es gestionar salones eventos programados mediante aprobaciones de solicitudes.

## Estándares

Se utiliza una plantilla de requerimientos (inspirada en el libro De S. Robertson y J. Robertson “Mastering the Requirements Process”, Addison-Wesley, 1999).

## Documentación

Los documentos enlazados a este tenemos documento tenemos:

* Casos de uso.
* Diagrama de casos de uso.
* Descripción del sistema.

## Definición de prioridades

**Alta:** Requerimientos relacionados a la definición y alineación de objetivos así como la evaluación y calificación de los mismos.

**Media:** Requerimientos relacionados con las consultas y monitoreo de las etapas de seguimiento.

**Baja:** Requerimientos de funcionalidades adicionales que facilitaran el uso del sistema.

* Requerimientos Software

Tomando en consideración características de Software de reservación, estudios realizados en aplicaciones con características similares y analizando los problemas detectados anteriormente, establecimos e identificamos distintos requerimientos indispensables para implementar en forma exitosa y segura una infraestructura para reservación de salones nombrada Resa (Reservador de salón ) los requerimientos fueron desglosados en requerimientos de negocio, usuarios, de sistema (Funcionales y no funcionales).

Esta sección se entregara un resumen de todos los requerimientos del software del sistema. Los requerimientos de negocio, de usuario y requerimientos del sistema.

A continuación se presenta la especificación inicial de requerimientos del sistema, el detalle de cada uno de ellos deberá ser analizado y documentado en la fase correspondiente de obtención de requerimientos de acuerdo al  Plan de Trabajo descrito más adelante en este documento y cumpliendo con los estándares especificados en el Plan de Calidad.

* Requerimientos de negocio.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Código:** | RN001 | **Tipo:** | Negocio |  |
| **Descripción:** | Desarrollar una aplicación de reservación de salones de eventos en la cual se pueda gestionar y administrar la agenda programada de cada salón. | | |

* Requerimientos de Usuarios.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID De Requisito** | **Descripción** | **Prioridad** | **Caso de uso R.** |
| RU001 | Para iniciar sesión se solicitara el código de empleado y la contraseña. | Alta | --------- |
| RU002 | En la pantalla principal deben verse los eventos programados en los salones. | Media | --------- |
| RU003 | Poder crear salones con sus respectivos inventarios y servicios. | Alta | -------- |
| RU004 | Poder Actualizar las características de los salones. | Alta | -------- |
| RU005 | Poder eliminar los salones. | Alta | -------- |
| RU006 | Que disponga de opciones para agregar, y eliminar solicitudes de eventos. | Alta | ------- |
| RU007 | Que disponga de una opción para visualizar el itinerario de cada salón. | Media | -------- |
| RU008 | El usuario con autorización de aprobación de solicitudes dispondrá de una opción accesible para aprobar. | Alta | --------- |
| RU009 | Se dispondrá de visualizar las solicitudes no aprobadas en la opción de gestión de las mismas. | Media | ------------ |
| RU010 | Al aprobarse una solicitud se enviaran un correo a los solicitantes. | Baja | ------- |
| RU011 | Se podrán eliminar los eventos ya programados en el sistema. | Alta | --------- |

* Requerimientos del sistema.

Este tipo de requerimientos son el resultado de acoplar los requerimientos de usuario al modelo de la solución. Estos ya tienen cierto nivel de detalle avanzado y son la base para empezar la fase de diseño del sistema de chat. Estos requerimientos hacen referencia hacia los requerimientos funcionales y no funcionales, los cuales están plasmados a continuación.

* Requerimientos Funcionales.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID Requisito** | **Descripción** | **Usuario** | **Prioridad** | **C.U Re.** |
| **RF001** | En la interfaz del se ejecutaran 3 hilos de verificación de conexión, verificación de archivos y carga de la progresive bar. | Todos | Alta | CU001 |
| **RF002** | En la interfaz de inicio de sesión el usuario no autentificado introducirá su id de empleado y contraseña para ingresar al sistema de reservación. | Todos | Media | CU002 |
| **RF003** | Como pantalla principal se mostrara un calendario Con los eventos más próximos de todos los salones. | Todos | Media | CU002 |
| **RF004** | El usuario dispondrá de una opción donde creara salones con las características concernientes. | Administrador | Alta | CU003 |
| **RF005** | El usuario dispondrá de una opción donde podrá actualizar las características de un salón. | Administrador | Alta | CU004 |
| **RF006** | El usuario dispondrá de una opción donde podrá eliminar salones. | Administrador | Alta | CU005 |
| **RF007** | Se agregaran solicitudes de eventos a respectivos salones. | Solicitante,  Administrador | Alta | CU006 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID Requisito** | **Descripción** | **Usuario** | **Prioridad** | **C.U Re.** |
| **RF008** | Se podrá eliminar solicitudes a los salones si estas aún no se a aprobado. | Solicitante,  Administrador | Media | CU007 |
| **RF009** | El usuario dispondrá de una opción de aprobación de solicitudes. | Administrador,  Coordinador, | Alta | CU008 |
| **RF010** | El sistema dispondrá de una opción de visualizar los salones con sus correspondientes eventos aprobados. | Todos | Media | CU009 |
| **RF011** | Cuando se apruebe una solicitud de evento se enviara un mensaje de aprobación al organizador. | Todos | Media | CU008 |
| **RF012** | El sistema dispondrá de una opción para eliminar los eventos. | Todos | Alta | CU010 |
| **RF013** | Generar de reportes conforme a la Normatividad Institucional de planeación. | Todos | Alta | CU012 |
| **RF014** | Cerrar la sesión en el sistema. | Todos | Media | CU011 |

Estudio de Factibilidad y Retorno de Inversión (ROI)

En el estudio de la Factibilidad, determinamos el presupuesto de costos de los recursos técnicos, humanos y materiales tanto para el desarrollo como para la implantación del Sistema. Además, nos ayudara a realizar el análisis costo-beneficio de nuestro sistema, el mismo que nos permitirá determinar si es factible a desarrollar económicamente el proyecto. A continuación se describe los costos del recurso necesario para el desarrollo de nuestro Sistema de Información:

|  |  |  |  |
| --- | --- | --- | --- |
| Recursos Humanos | | | |
| Nº | Cargo | Costo Individual | Costo Total |
| 1 | Desarrollador | 1500 x hora (8 horas) | $RD120,000 |
| 2 | Programadores | 1000 x hora (8 horas) | $RD80,000 |
|  | | **Total:** | $RD200,000 |

|  |  |  |  |
| --- | --- | --- | --- |
| Recursos Tecnológicos | | | |
| Hardware | | | |
| Nº | **Descripción** | **Costo** | **Costo Total** |
| 1 | Computadora de escritorio. | $RD8000 | $RD8000 |
| Software | | | |
| 1 | Sistema Operativo | $RD3500 | $RD3500 |
|  | | **Total:** | $RD11500 |

|  |  |
| --- | --- |
| Flujo de pago | |
| Recursos | **Costos** |
| Recursos Humanos | $RD120000 |
| Recursos Tecnologicos | $RD11500 |
| Total: | $RD131,500 |