Propuesta de solución a la Vista de Arquitectura de Datos del Sistema GESPRO 12.05

Trabajo final presentado en opción al título de

Ingeniero en Ciencias Informáticas

Facultad 4

Autor: Luis Angel López Zaldivar

Tutor: Ing. Yusimy Rodríguez Ruiz

La Habana, junio del 2012

PENSAMIENTO

"La vida es una obra de teatro que no permite ensayos... Por eso canta, ríe, baila, llora, y vive intensamente cada momento de tu vida... Antes que el telón baje y la obra termine sin aplausos"

Charles Chaplin

"Si ríes, el mundo reirá contigo; si lloras, el mundo dándote la espalda te dejará llorar".

Charles Chaplin

Agradecimientos

Que iba a pensar que esta es una de las partes más difícil del trabajo; “Agradecimientos”, una palabra que envuelve a muchas personas involucradas que hicieron posible que yo realizara este trabajo de diploma para que me graduase con éxito de Ingeniero en Ciencias Informáticas, un sueño hecho realidad. Son tantas las personas que hicieron posible que yo realizara este trabajo que le doy un agradecimiento general para que no se me quede ninguno de ellos. Pero en especial quiero agradecerle a:

En primer lugar quiero agradecerle a “Dios” porque ha estado presente en cada paso que he dado en esta escuela, ayudándome a pasar cada año que he cursado en esta Universidad.

Quiero darle un agradecimiento especial a Yoraima Peña Hinojosa, que ella es la persona que hizo posible gracias a sus esfuerzos extras, y soportando mis dolores de cabeza durante 3 años, que hoy llegara a la discusión de este trabajo, gracias a su apoyo incondicional y al amor que me brindó durante todo este tiempo.

Agradecerles a mi Mamá y mi Papá, darle todas las gracias por su apoyo en la realización de este sueño, y espero poder ser el orgullo que tanto soñaron ellos.

No se me puede quedar un agradecimiento de corazón a mi abuelita Miriam, que tanto luchó con mis padres para que yo obtuviera sus apoyos en mis estudios.

A mi tía Leticia y mi tío José, que tanto me apoyaron y me ayudaron en esta difícil carrera.

Un agradecimiento a mi Familia linda que me supo apoyar en cada momento:

1. Mi abuelo Pablo
2. Mi tía Chachi
3. A mis hermanos y primos
4. Mi tío Orestes, a Jorge, a Nuria

que también pusieron su granito en esta ardua tarea.

AGRADECIMIENTOS

Un “Agradecimiento” muy especial a mi tutora Yusimy Rodríguez Ruiz que sin su ayuda no hubiera hecho posible la realización de este trabajo final. Gracias por sus exigencias profe, gracias por su apoyo en los momentos más difícil de este trabajo. Muchas gracias de todo corazón.

También a los profesores del proyecto de GESPRO que me ayudaron en todo momento: a Pedro, Félix, entre tantos otros que no recuerdo los nombres en este momento. Pero gracias a todos.

Quiero agradecerles a mis compañeros y hermanos de universidad, que me apoyaron en todo momento.

Agradecerle a la Revolución y a Fidel por permitir y darme la posibilidad de hacerme Ingeniero en Ciencias Informáticas.

“A todos Mucha Gracias de todo corazón”.

Dedicatoria

Esta es la otra sección más difícil del trabajo de diploma, porque son tantas las personas que hicieron posible que yo me graduara de Ingeniero, que por tal razón a las cuales les dedico este trabajo final.

En primer lugar y la dedicatoria especial a la persona por la cual yo llegue a ser alguien en esta vida; si se puede decir que lo soy. La persona que me inculco y creó en mí los valores que tengo hoy, que sin su educación, su cariño y amor yo no sé que hubiera sido de mi vida. Para ti va dedicado este trabajo, para ti mi abuelita Rosaida, tu no sabe cuánto siento que tu no me hubieras visto recibir el titulo por el cual tanto luchaste conmigo. El titulo que tanto querías tú que yo recibiera en mi vida. La vida es dura y me privo de la posibilidad de que tú me vieras en este momento de mi vida, pero donde quiera que estés, yo se que estás orgullosa de mi, porque no te defraudé. Y por tal razón para ti va dedicada mi tesis, mi título, en especial va dedicada mi vida. Todo lo que he hecho y hare siempre va a ser pensando en lo que querías tu que yo fuera.

También quiero dedicarles este trabajo y este título a mis padres queridos, para ti mami y para ti papi, que tanto se sacrificaron para verme hecho ingeniero.

A mi abuelita Miriam, a mi abuelo Pablo, a mis tías Lety y Clara.

A mis hermanitos y a mis primos Leo y María Clara, que espero que ellos también algún día puedan ser el orgullo de sus padres.

A mi familia en general que tanto me apoyaron, no pongo mas nombres porque sería injusto si se me queda alguno.

Les dedico este trabajo de diploma a todos ustedes, que hicieron que este sueño se hiciera realidad, dándome su apoyo incondicional.

Declaración Jurada

Declaro por este medio que yo Luis Angel López Zaldivar, con carnet de identidad 88021226269, soy el autor principal del trabajo final de tesis **Propuesta de solución a la Vista de Arquitectura de Datos del Sistema GESPRO 12.05**, y que autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Y para que así conste, firmo la presente declaración jurada de autoría en La Habana a los 27 días del mes de junio del año 2012.

_____	_____
Nombre del Autor	**Nombre del Tutor**
Luis Angel López Zaldivar.	**Ing. Yusimy Rodríguez Ruiz**

RESUMEN

La arquitectura en cualquier software del mundo es de gran importancia, debido a que con esta se define el funcionamiento de cualquier sistema. En Cuba la Universidad de las Ciencias Informáticas (UCI) constituye una institución que lleva la avanzada en la elaboración de productos informáticos. En este centro se construyó un sistema (GESPRO[1]) para garantizar la gestión de los productos desarrollados en la UCI; el mismo se encuentra en su versión 11.05. Debido a que se han detectado deficiencias que no permiten el correcto desempeño de sus funcionalidades, se ha decidido implementar una nueva versión. El objetivo de esta investigación fue definir e implantar la vista de la arquitectura de datos de la suite de GESPRO en su versión 12.05, con el fin de corregir las dificultades presentes en la versión anterior. Para darle cumplimiento al objetivo planteado se utilizó el modelo 9+1 vista de la arquitectura propuesto por la UCI, definiendo los escenarios por los cuales van a transitar los datos de los proyectos. Haciendo uso de la herramienta de modelado Visual Paradigm se diseñó el modelo de una base de datos capaz de almacenar la información vinculada a los proyectos que cuenta con la definición de un diccionario de datos para evitar ambigüedades a la hora de insertar los datos en el sistema. Se validó la solución mediante el método Delphi demostrando que cumplía la calidad necesaria para ser implantada.

Palabras Claves: gestión de proyectos; diccionario de datos, modelo de datos; vista de arquitectura de datos.

[1] *Sistema de Gestión de proyectos de la Universidad de las Ciencias Informáticas*

ÍNDICE DE CONTENIDOS

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

En la actualidad con el vertiginoso desarrollo de las tecnologías de la información y las comunicaciones se ha incrementado en todo el mundo el desarrollo de software. Uno de los motivos principales que ha dado cabida a este desarrollo es el de gestionar e integrar la información en aras de mejorar los procesos del negocio.

El aumento de los centros productores de software en el mundo ha traído consigo que la gestión de los proyectos se haga más rigurosa para mantener de una manera organizada todo lo relacionado con la documentación, los medios de los proyectos y la información necesaria que se maneja.

El autor Ernest Abadal Falgueras define a la gestión de proyectos como: *"una metodología muy conocida y utilizada que sirve para facilitar el diseño, la planificación y la ejecución de productos y servicios de todo tipo y de todos los sectores. Los profesionales de la información y documentación recurren cada vez más a la gestión de proyectos para la creación de nuevos servicios…"* **(40)**

La Universidad de las Ciencias Informáticas es quien va en la avanzada de la producción de software en Cuba. Esta sobre el año 2009 contaba con un conjunto de centros de desarrollo de software y polos productivos que usaban para la gestión de los proyectos distintas herramientas. Las mismas eran fundamentalmente: Trac, Redmine, Dot Proyect, Jira, Microsoft Proyect, Egroupwere[2] y fueron utilizadas hasta el año 2010.

El empleo de diversas herramientas de gestión de proyectos trajo como consecuencia que el trabajo se tornara engorroso pues contaban con diferentes bases de datos y las conexiones de estas con las herramientas de gestión eran sumamente lentas.

Por tal motivo, y en paralelo con la reorganización del área productiva de la universidad en centros de desarrollo de software, se decide crear un único sistema que guiara el trabajo que se realiza hacia una misma metodología de gestión, y que todas las bases de datos se integren en una sola. Surge así GESPRO que es una herramienta para la gestión de proyectos que se encuentra hoy en su versión 11.05.

[2] Sistemas de gestión de proyectos.

Tras una encuesta realizada (**Anexo 1**) a los especialistas en gestión de proyectos en la universidad acerca de la funcionalidad del sistema y de la gestión de los datos de la herramienta en la base de datos de GESPRO, se detectaron un conjunto de insuficiencias.

Entre ellas está que la versión actual de GESPRO presenta inconveniencias en diferentes escenarios[3] que se encuentran en la vista de datos, ya que estos no cuentan con una definición precisa de un patrón de solución, por lo que el funcionamiento de la base de datos del GESPRO no es lo suficientemente eficiente. Otra irregularidad encontrada es con respecto al diccionario de datos[4], ya que la base de datos no cuenta con una correcta definición de los términos. El modelo de datos correspondiente a la base de datos del GESPRO no se ha diseñado, por lo que las relaciones entre las tablas no se tienen correctamente representadas y esto provocaría que el funcionamiento de la base de datos no sea lo más óptimo posible.

Dado todas estas deficiencias encontradas en la versión 11.05 del sistema de gestión de proyecto en la UCI, surge la idea de realizar una mejora (versión 12.05) que elimine todos los inconvenientes encontrados en la encuesta realizada a los especialistas de esta área productiva de la Universidad de las Ciencias Informática.

A partir de la problemática planteada surge como **problema a resolver** que *las insuficiencias en la arquitectura de datos de GESPRO está afectando la gestión de los datos a diferentes niveles de decisión.*

Para llevar a cabo dicha investigación se plantea como **objeto de estudio** la arquitectura de datos y como **objetivo general** definir e implantar la vista de arquitectura de datos de la suite GESPRO 12.05.

La investigación tiene como **campo de acción** la arquitectura de datos en el sistema de gestión GESPRO.

Los **Objetivos específicos** para resolver el problema planteado son los siguientes:

- Elaborar el marco teórico de la investigación.
- Realizar una propuesta de solución de la vista de arquitectura de datos de GESPRO 12.05.
- Validar la propuesta de solución realizada.

[3] *Describen las secuencias de acciones a realizar por un sistema informático.*
[4] *Es donde se tienen especificados los términos de la base de datos.*

Para darle cumplimiento a los objetivos específicos planteados se definieron las siguientes **tareas de investigación**:

- Elaboración del marco teórico de la investigación.
- Descripción de los escenarios existentes en la vista de datos.
- Determinación de los patrones de solución para cada uno de los escenarios descritos.
- Confección del diccionario de datos.
- Construcción del modelo de datos.
- Validación de los resultados alcanzados.

Para la realización de este trabajo se utiliza el **tipo de investigación** explicativa, y se defiende la **hipótesis** de que *"si se define e implanta la vista de arquitectura de datos basado en el modelo de la guía base de la arquitectura en la Universidad de las Ciencias Informática, entonces se permitirá la gestión de los datos a diferentes niveles de decisión en la gestión de proyecto"*.

Teniendo en cuenta la hipótesis planteada se define como **variable dependiente** la gestión de los datos a diferentes niveles de decisión en la gestión de proyecto.

Se toma como **población** a estudiar, los centros de la red de producción de la Universidad de las Ciencias Informáticas y como **muestra** la red de centros de la sede central.

Para el **diseño de la investigación** se utiliza el método pre experimental y como **análisis estadístico** la validación en casos de estudios en la red de centros.

Para el desarrollo de la investigación se emplearon los siguientes métodos de investigación científica e instrumentos.

Métodos

Métodos teóricos:

- Histórico-lógico: para determinar las tendencias actuales de desarrollo de los modelos y enfoques arquitectónicos.
- Sistémico: para determinar los componentes y definir las relaciones entre estos.

- Analítico-sintético: para procesar la información y arribar a las conclusiones de la investigación, así como para precisar las características del modelo arquitectónico propuesto.

Métodos empíricos:

- Observación: este método se empleó para la percepción selectiva de las restricciones y propiedades del sistema.
- Entrevista: este método sustenta en gran medida la investigación ya que la mayor parte de la información está en manos de los especialistas.
- Encuesta: se utilizó en la validación de la propuesta de solución de la investigación.

Instrumentos

- Entrevistas
- Encuestas
- Evaluaciones

Con la realización e implantación de la arquitectura de la vista de datos del sistema GESPRO 12.05 se espera como **aporte práctico** facilitar el trabajo con los datos a diferentes niveles de decisión en la gestión de proyecto, y elaborar el diccionario de datos y modelo de datos, de la base de datos del GESPRO 12.05.

La investigación se encuentra estructurada en tres capítulos:

Capítulo 1: Fundamentación teórica: En este capítulo se muestra todo lo concerniente a la teoría de la investigación. En él se analizaron los conceptos de Arquitectura de Software[5], las diferentes escuelas de la arquitectura, sus estilos y patrones, así como sus vistas arquitectónicas. Se consideraron diferentes herramientas para la gestión y modelación de bases de datos. Por último a través de las conclusiones se precisan los elementos que se tuvieron en cuenta para la propuesta de solución.

Capítulo 2: Diseño arquitectónico de la vista de datos del GESPRO 12.05: Este capítulo muestra la propuesta de solución dado al problema planteado en la introducción de la investigación. Se definieron y describieron los escenarios de la base de datos del sistema, con sus patrones de solución y los atributos

[5] *Es la que define como se debe elaborar un sistema informático.*

de calidad que se mejoran con la puesta en práctica de la solución propuesta para cada escenario. Se elaboró un diccionario de datos con la definición de cada término que intervienen en la base de datos del GESPRO, así como el modelo de datos de la misma.

Capítulo 3: Validación de la arquitectura propuesta para la vista de datos del GESPRO 12.05: En este capítulo se evaluó la arquitectura propuesta. Se utilizó el método Delphi para realizar la validación de la vista de la arquitectura de datos, demostrando la calidad de la solución planteada.

Capítulo 1. Fundamentación teórica

En la actualidad para la construcción y el desarrollo de los productos informáticos, se hace necesaria e imprescindible la definición de una buena arquitectura de software. Esta es la columna vertebral de un sistema o producto informático y en ella se basa el buen funcionamiento del mismo. Los primeros pasos en la arquitectura de software datan de los años 1960, donde ya se empezaba a definir el concepto de *arquitectura de software* en los círculos de investigación. La Arquitectura de Software al igual que los planos de un edificio, estas indican la estructura, funcionamiento e interacción entre las partes del software. Toda arquitectura debe describir diversos aspectos del sistema, donde especifique los pasos, conceptos, tratamientos y funcionalidad de las partes del sistema.

En este capítulo se abordan elementos teóricos esenciales que se relacionan con la arquitectura de software. Se analizan sus definiciones así como los estilos, patrones y vistas de la arquitectura. Además se hace referencia a conceptos como: diccionario de datos, escenarios, arquitectura de datos ya que la investigación está basada en estos elementos de la arquitectura.

1.1 ARQUITECTURA DE SOFTWARE

La arquitectura de software es definida por diferentes autores y especialistas de esta rama. A continuación se muestran algunos de los criterios más conocidos.

El **IEEE[6] Std 1471-2000** define a la arquitectura de software como:" *la organización fundamental de un sistema incorporada en sus componentes, en sus relaciones mutuas y en el entorno, y los principios que guían su diseño y evolución."* **(7)**

En el libro **Arquitectura de Software en la Práctica** (**Software Architecture in Practice) 2nd edición se plantea que** la arquitectura de software es "*la estructura de estructuras de un sistema, la cual abarca componentes de software, propiedades externas visibles de estos componentes y sus relaciones*". **(8)**

El ingeniero de software Roger S. Pressman[7] define en sus escritos que:

"La arquitectura no es el software operacional, más bien es la representación que capacita al ingeniero del software para: analizar la efectividad del diseño para la consecuencia de los requisitos fijados, considerar

[6] *Siglas en ingles del Instituto de Ingenieros Eléctricos y Electrónicos.*
[7] *Ingeniero de software, escritor y consultor, norte-americano, presidente de R.S. Pressman & Associates.*

las alternativas arquitectónicas en una etapa en la cual hacer cambios en el diseño es relativamente fácil y reducir los riesgos asociados a la construcción del software". **(9)**

De manera general, la arquitectura de software es quien define el diseño de un sistema de software, así como su desarrollo. Una buena definición de la arquitectura de software determina una buena construcción del producto en cuestión.

La arquitectura de software se encuentra organizada en diferentes formas definidas por distintas corrientes o escuelas estudiosas del tema en cuestión. Estas escuelas emplean estilos y patrones arquitectónicos en función de la complejidad del producto o sistema a diseñar. Así mismo, algunas guían el trabajo a través de vistas arquitectónicas que poseen características propias que permiten lograr un mejor diseño de la arquitectura.

A continuación se expone una panorámica general de las escuelas más representativas en el tema de arquitectura de software.

1.2 ESCUELAS O CORRIENTES REPRESENTATIVAS DE LA ARQUITECTURA DE SOFTWARE

En el campo de la arquitectura de software, se destacan seis corrientes representativas de la arquitectura de software. Las mismas son:**(24) (25)**

1. Arquitectura estructural, basada en un modelo estático de estilos.
2. Estructuralismo arquitectónico radical
3. Arquitectura basada en patrones
4. Arquitectura procesual
5. Arquitectura basada en escenarios.
6. Arquitectura como etapa de ingeniería y diseño orientada a objetos

Arquitectura estructural, basada en un modelo estático de estilos

Constituye la corriente clásica de la disciplina. Los representantes de esta corriente son todos académicos, mayormente de la Universidad Carnegie Mellon en Pittsburgh. En toda la corriente, el diseño arquitectónico no sólo es el de más alto nivel de abstracción, sino que no tiene por qué coincidir con la configuración explícita de las aplicaciones; rara vez se encontrarán referencias a lenguajes de programación o piezas de código **(28) (29)**.

Estructuralismo arquitectónico radical

En este movimiento hay al menos dos tendencias, una que excluye la relevancia del modelado orientado a objetos para la arquitectura de software y otra que procura definir nuevos metamodelos y estereotipos de UML[8] como correctores de la situación. UML 2.0 constituye el principal ejemplo de esta tendencia **(30)**. Esta tendencia es un derivado de la escuela anterior, y predomina fundamentalmente en los países de Europa.

Arquitectura basada en patrones

Esta escuela está basada principalmente en la redefinición de los estilos como patrones arquitectónicos o de diseño **(31) (32)**. El diseño consiste en identificar patrones preexistentes, que se definen en forma parecida a los estilos de arquitectura, con características de solución a problemas arquitectónicos comunes y clasificados. El volumen de soluciones de tipo presentes, resulta el punto de referencia incorporado en la propuesta de solución, con el objetivo de garantizar la unicidad de soluciones y en la posibilidad de la comunicación entre arquitectos y el propio equipo de desarrollo.

Arquitectura procesual

Intenta establecer modelos del ciclo de vida y técnicas de diseño de requerimientos, diseño, análisis, selección de alternativas, validación, comparación, estimación de calidad y justificación económica, específicas para la arquitectura de software. Otras variantes dentro de la corriente procesual se caracterizan por cambios en la etapa del proceso de extracción de la arquitectura, generalización y reutilización **(25)**.

Arquitectura basada en escenarios.

[8] *Siglas en ingles del Lenguaje Unificado de Modelado*

La Arquitectura basada en escenarios es la corriente más nueva de todas las mencionadas. Se trata de un movimiento mayormente europeo con centro en Holanda. Recupera el vínculo de la arquitectura de software con los requerimientos y la funcionalidad del sistema, que se encuentran algo borroso en la arquitectura estructural clásica. En esta corriente suele utilizarse diagramas de casos de uso UML como herramienta, dado que los casos de uso son uno de los escenarios posibles y que no están orientados a objetos. Los autores que trabajan con esta corriente son, aparte de los codificadores de ATAM, CBAM, QASAR y demás métodos del SEI, los arquitectos holandeses de la Universidad Técnica de Eindhoven, de la Universidad Brije, de la Universidad de Groningen y de Philips Research. La participación de los holandeses en esta corriente arquitectónica es considerable; la Universidad de Eindhoven es, el lugar en el que surgió lo que P. I. Sharp planteaba llamar la escuela arquitectónica de Dijkstra[9] **(33)**.

Arquitectura como etapa de ingeniería y diseño orientada a objetos

Esta corriente está basada en el modelo de James Rumbaugh[10], Ivar Jacobson[11], Grady Booch[12], Craig Larman[13], relacionando a UML con Rational **(26)**. En esta referencia la arquitectura se restringe, enfocándose en los niveles más elevados de abstracción. Importa más la abundancia, el detalle de diagramas y técnicas disponibles, que la visión del conjunto.

La definición de arquitectura que se inicia en esta corriente tiene que ver con aspectos formales a la hora del desarrollo. Las definiciones revelan que la Arquitectura de Software, en este aspecto, corresponde a decisiones sobre organización, selección de elementos estructurales, comportamiento, composición y estilo arquitectónico susceptibles de ser descriptas a través de las cinco vistas clásicas del modelo 4+1 de Kruchten[14] **(27)**.

Analizando las corrientes arquitectónicas se decide escoger la escuela: *arquitectura como etapa de ingeniería y diseño orientada a objetos*; que contempla el modelo 4+1 vistas para realizar una comparación con un modelo nuevo planteado por especialistas de la Universidad de las Ciencias Informáticas que contemplan 9+1 vistas de la arquitectura y llegar a conclusiones precisas acerca de qué modelo es el más indicado para usar en la investigación.

[9] *Científico holandés que dedico su vida a las ciencias.*
[10] *Científico de la computación y un metodologista de objeto.*
[11] *Ingeniero sueco en Ciencias de la computación.*
[12] *Es director científico de Rational Software (ahora parte de IBM).*
[13] *Científico canadiense en ciencias de la computación.*
[14] *Ingeniero de Software canadiense.*

1.3 VISTAS ARQUITECTÓNICAS

Las vistas arquitectónicas en la arquitectura de software permiten visualizar, entender y razonar los elementos más significativos en ella y a su vez identificar las áreas de riesgo que requieren mayor detalle de elaboración. Toda arquitectura de software debe representar diversos aspectos. Si se utilizan distintos modelos o vistas se puede describir de manera más comprensibles estos aspectos.

En la arquitectura de software mientras la estructura y diseño lo exijan se puede contar con tantas vistas como sean necesarias independientemente de cualquier metodología existente. Las vistas usadas para una arquitectura dependen de las necesidades y el uso que se le dará a la misma.

A continuación se da una panorámica y descripción general de las vistas propuesta por el modelo 4+1 vista, que Utiliza la metodología RUP (Rational Unified Process) entre otras.

1.3.1 Vistas propuestas en el modelo 4+1

Una de las metodologías de desarrollo de software más utilizada en la industria de software es RUP la cual utiliza un modelo de 4+1 **(ver figura 1)** vista para representar la arquitectura de software: Estas vistas son las siguientes: **(5)**

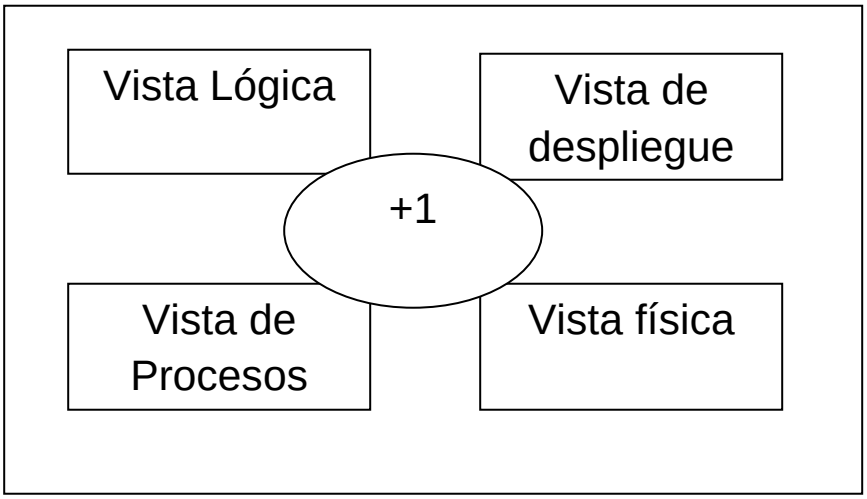

Figura 1. Vistas del modelo 4+1 vista

Vista de casos de usos:

Lista los casos de uso o escenarios del modelo de casos de uso que representen funcionalidades centrales del sistema final, que requieran una gran cobertura arquitectónica o aquellos que impliquen algún punto especialmente delicado de la arquitectura.

Vista lógica:

Describe las partes arquitectónicamente significativas del modelo de diseño, como son la descomposición en capas, subsistemas o paquetes, y una vez presentadas estas unidades lógicas principales, se profundiza en ellas hasta el nivel que se considere adecuado.

Vista de procesos:

Describe la descomposición del sistema. Indica qué procesos o grupos de procesos se comunican o interactúan entres sí y los modos en que estos se comunican.

Vista de despliegue:

Describe uno o más escenarios de distribución física del sistema sobre los cuales se ejecutará y hará el despliegue del mismo. Muestra la comunicación entre los diferentes nodos que componen los escenarios antes mencionados, así como el mapeo de los elementos de la Vista de Procesos en dichos nodos.

Vista de implementación:

Describe la estructura general del modelo de Implementación y el mapeo de los subsistemas, paquetes y clases de la Vista Lógica a subsistemas y componentes de implementación.

Ya analizado el modelo 4+1 vistas arquitectónica, se describirá a continuación el modelo propuesto por especialistas de la Universidad de las Ciencias Informáticas que contempla 9+1 vistas de la arquitectura de software.

1.3.2 Modelo de vistas propuesto por la Universidad de las Ciencias Informática

Dada la necesidad de definir la arquitectura de software en los productos informáticos que se desarrollan en la Universidad de las Ciencias Informática (uno de los centros productores de software en Cuba), se hizo necesario de plantear y avalar un modelo de arquitectura que contemplara algunas vistas que no se tienen en cuenta en el modelo 4+1, ya que se hacen necesarias a la hora de desarrollar la arquitectura de un sistema.

Las vistas que contemplan el modelo planteado por los especialistas de la Universidad de las Ciencias Informática (UCI) guían el trabajo en dos objetivos fundamentales (que hacer y cómo hacerlo). Entre los

que dicen que hacer en la arquitectura de un producto informático se encuentran las vistas de procesos, presentación, datos, sistema y seguridad; y entre las que dicen como se debe aplicar se encuentran las vistas de integración, tecnología, despliegue e infraestructura. También este modelo de vistas contempla una Guía Base[15], que conforman junto a las nueves vistas antes planteada el modelo 9+1 vista, que así es como se le denomina a este modelo planteado por especialistas de la UCI **(34) (Ver figura 2).**

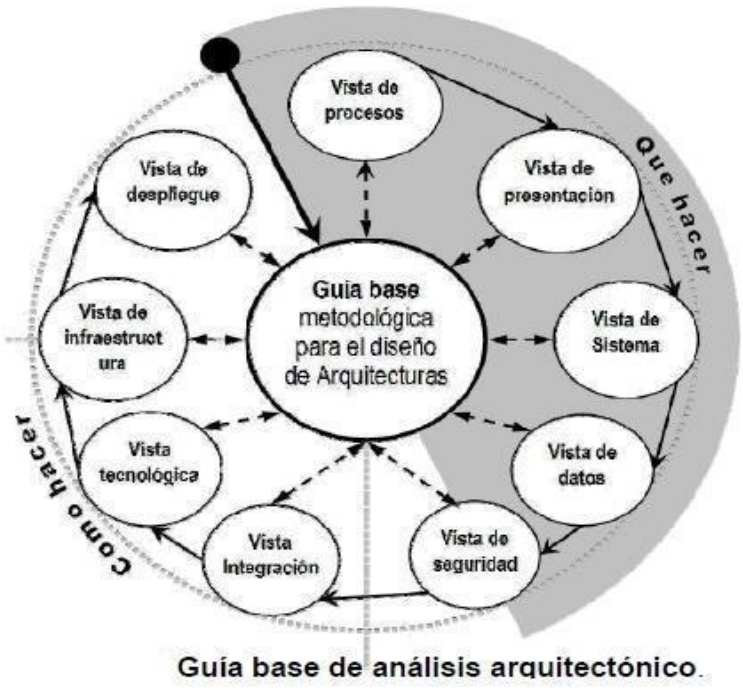

Figura 2. Vistas que propone el modelo 9+1 vistas

A continuación se dará una panorámica de los objetivos fundamentales que contempla cada una de estas vistas, así como su funcionalidad en la construcción del sistema.

Vista de procesos:

La vista de procesos es la que se encarga de la caracterización del entorno según la vista donde se debe identificar el dominio de la solución propuesta, también se debe conceptualizar los macros de procesos del entorno corporativo, aquí se identifican las áreas de procesos, y procesos concretos, clasificándolos en

[15] *Se le denomina así al modelo de vistas de la arquitectura (9+1) planteado por la Universidad de las ciencias Informáticas.*

claves, estratégicos y de soporte. Se diseña un mapa conceptual por procesos y se realiza la priorización de los procesos del negocio.

Vista de presentación:

En la vista de presentación se debe de tener en cuenta los siguientes elementos como son la definición de pantallas que contempla tres tipos fundamentales

- ✓ Pantalla tipo genérica
- ✓ Pantalla de bienvenida
- ✓ Pantalla acceso a módulos (Opcional)

Vista de Sistema:

La vista de sistema en el modelo planteado por la Universidad de las Ciencias Informática contempla 7 aspectos fundamentales. Estos aspectos están relacionados con la caracterización del producto según la vista donde se realiza una caracterización atendiendo a los elementos:

- Personal de desarrollo
- Dinamismo
- Cultura
- Tamaño del equipo
- Criticidad

Otros aspectos que se definen en la vista de sistema son los que tienen que ver con la priorización de requisitos funcionales del sistema (Complejidad, relevancia para el negocio e impacto en la arquitectura), también la representación de la dependencia entre paquetes y la descripción de los escenarios y patrones de solución así como el agrupamiento en subsistemas o paquetes y priorización de los requisitos priorizados, la identificación de los paquetes y componentes principales, la representación de la dependencia entre paquetes tanto en paralelo como en sucesión, son los aspectos que contemplan dicha vista arquitectónica.

Vista de Seguridad:

En los productos informáticos la seguridad es un tema que tiene que estar presente, ya que sin ella los sistemas serian vulnerables a las redes, donde se mueven gusanos informáticos, virus e intrusos que se dedica a jaquear los proyectos para obtener y robar la información de los mismos.

Por tal razón en el modelo planteado por la UCI se incluye la vista de seguridad y esta contempla las especificaciones de seguridad:

- Datos: Donde se define la seguridad de los datos y el acceso a los mismos.
- Código: Donde se especifica el estado del código (ofuscado y sellado, así como las tecnologías usadas para esto).
- Nodos de despliegue e integración.
- Capa de presentación y entorno de trabajo.

Vista de Integración:

Otra vista agregada al modelo antes mencionado es la vista de integración donde esta contempla 4 aspectos fundamentales, uno de estos aspectos es la selección de los estilos arquitectónicos a emplear, también está presente la selección de la estrategia de diseño y modelo de desarrollo a utilizar, así como la selección de patrones de diseño principales, vista desde la integración entre las soluciones y la selección de los estándares de codificación.

Vista Tecnológica:

En la vista tecnológica se definen un aspecto fundamental donde se selecciona la tecnología a usar. Esta tecnología puede ser definida especificando la clase o el tipo y también especificando si es candidata o propuesta.

Vista de Despliegue:

Esta vista es la encargada de contemplar todo lo relacionado a donde se va a montar el sistema en cuestión. Ella cuenta con los siguientes aspectos a tener a desarrollar.

- Caracterización de la solución vista desde las licencias de los componentes que la forman: Donde se seleccionan los tipos de licencias que soportan los componentes de la solución.
- Caracterización de la solución vista desde las estrategias de desarrollo empleadas para su elaboración: Donde se seleccionan las estrategias de desarrollo empleadas.
- Definición del tipo de licenciamiento de la solución vista desde la arquitectura: Donde se selecciona la estrategia de licenciamiento a utilizar.
- Definición de las estrategias de ingreso vista desde la arquitectura: Donde se selecciona la estrategia para la generación de ingresos.
- Defina la estrategia de empaquetamiento del producto para la implantación.
- Declare los requerimientos mínimos de instalación.
- Declarar la estrategia para el Soporte y los detalles de los niveles de soporte que se van a realizar.

Vista de Infraestructura

La vista de infraestructura en el modelo realizado por la UCI plantea 4 aspectos fundamentales para describir y definir la vista. Los aspectos que contempla dicha vista son los siguientes:

- Describir el diagrama de configuración de redes para el cual está previsto el despliegue de la solución.
- Describir el diagrama de configuración del software.
- Describir el diagrama de configuración de hardware.
- Describir los escenarios y patrones de solución para la vista de infraestructura.

Vista de Datos:

Describe los elementos principales del Modelo de Datos, brindando un panorama general de dicho modelo en términos de tablas, vistas, índices. Esta vista es la encargada de controlar el panorama de la información y datos de un sistema. Se divide en 4 puntos fundamentales que contempla el tratamiento de los datos en las bases de datos de los sistemas informáticos. En esta vista se tienen en cuenta la

caracterización del producto según la vista (Donde se identifican los diferentes tipos de datos que se desean almacenar), se describen los escenarios y patrones de solución para cada uno de los escenarios existentes en la vista, se desarrolla un diccionario de datos donde se relacionan en esta sección todos los elementos necesarios para identificar los conceptos que influyen significativamente en la modelación de la base de datos y por último se construye el modelo de datos que recoge las especificaciones del diseño lógico y físico de la solución, es decir de la base de datos.
Esta vista que se presenta es de gran importancia tratarla. Ella es la que regirá el capítulo 2, ya que se define e implanta la vista de datos del sistema de GESPRO.

Analizado el modelo propuesto por la Universidad de la Ciencias Informática y el modelo 4+1, se percata que para la elaboración y construcción de un sistema informático el modelo 9+1 vistas es más completo para definir la arquitectura de software. Contempla vistas que en otros modelos no están contempladas, entre las vistas que contempla este modelo esta la vista de datos que es el punto de partida de la investigación. Por todo lo expuesto anteriormente se decide realizar la investigación basándose en el modelo 9+1 vistas de la arquitectura.

1.4 ESTILOS ARQUITECTÓNICOS

Los estilos arquitectónicos unen los componentes que ejecutan una función requerida por el sistema e influyen en los conectores que hacen la comunicación y cooperación entre dichos componentes.

En un software se pueden encontrar numerosos estilos arquitectónicos, algunos de ellos son los siguientes: **(3)**

- Arquitecturas basadas en componentes
- Arquitectura Modelo Vista Controlador (MVC)
- Arquitectura en capas
- Arquitecturas Orientadas a Objetos

1.4.1 Arquitecturas basadas en componentes

Los sistemas basados en este estilo arquitectónico se fundamentan en principios definidos por una ingeniería de software específica, por lo que las unidades de modelado, diseño e implementación de estos son los componentes. La arquitectura basada en componentes permite su restructuración. Los componentes soportan algún régimen de introspección[16], de modo que su funcionalidad y propiedades puedan ser descubiertas y utilizadas en tiempo de ejecución

Entre algunas ventajas que presenta este estilo arquitectónico se encuentran: **(1)**

- Las pruebas pueden ser ejecutadas probando cada uno de los componentes antes de que se pruebe todo el conjunto de componentes ensamblados.
- La calidad de una aplicación basada en componentes mejora con el tiempo dado que un componente puede ser construido y mejorado seguidamente por un experto u organización.
- Permite una mayor reutilización del software.
- Simplifica el mantenimiento del sistema ya que cuando existe un débil acoplamiento entre componentes, el desarrollador es libre de actualizar y/o agregar componentes según sea necesario, sin afectar otras partes del sistema.

Sus principales desventajas radican en que si los componentes no existen hay que crearlos y se pierde mucho tiempo; y que los desarrolladores no cuentan con las actualizaciones de los componentes que se adquieran. **(1)**

1.4.2 Arquitectura Modelo Vista Controlador (MVC)

Este estilo es utilizado mayormente cuando se necesita modularizar la interfaz de usuario, el control de eventos y las reglas de negocio. Tanto la vista como el controlador dependen del modelo, mientras que este no depende de las otras clases. Dicha separación permite realizar y probar el modelo, independientemente de la representación visual. Todo esto también propicia que la interfaz de usuario pueda mostrar, simultáneamente, múltiples vistas de los mismos datos; al igual que la adaptación al cambio puesto que los requerimientos de interfaz de usuario tienden a cambiar con mayor rapidez las reglas del negocio.

[16] *Observación y examen que una persona hace de sus propias ideas, pensamientos y sentimientos.*

Modelo: Es quien administra los datos del dominio de aplicación y el comportamiento. Mantiene el conocimiento del sistema. No depende de ninguna vista y de ningún controlador.

Vista: Es la encargada de mostrar toda la visualización de la información.

Controlador: Interpreta toda la información y acción que se realice tanto con el teclado como con el mouse, le informa al modelo y a la vista para que estas cambien según la información recibida. Presenta tres variantes fundamentales: Activa, Pasiva y Documento-Vista.

1.4.3 Arquitectura en capas

Este es unos de los estilos que aparecen con mayor frecuencia. Garlan y Shaw[17] definen que el estilo en capas es una organización jerárquica donde cada capa le brinda servicios a la capa inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior. En este estilo los componentes son las capas o niveles, que pueden estar implementadas internamente tanto por objetos como por procedimientos. Cada nivel tiene asociada una funcionalidad: los niveles bajos implementan funciones simples, ligadas al hardware o al entorno, mientras que los niveles altos implementan funciones más abstractas. **(1)**

1.4.4 Arquitecturas Orientadas a Objetos

Los componentes de este estilo son los objetos, o más bien instancias de los tipos de datos abstractos. Los objetos representan una clase de componentes denominada manager (administradores) debido a que son responsables de preservar la integridad de su propia representación. Un rasgo importante de este aspecto es que la representación interna de un objeto no es accesible desde otros objetos. **(2)**

Cada uno de los estilos arquitectónicos que se mencionan anteriormente presenta sus beneficios y debilidades. Escoger uno u otro depende de las necesidades y servicios del sistema que se quiere realizar. Siempre se debe seleccionar un estilo según las características del sistema y la arquitectura a implantar en el mismo.

[17] *Ingenieros en ciencias de la computación.*

1.5 PATRONES DE ARQUITECTURA

Con los patrones de diseño se puede construir la práctica colectiva de Ingeniería de Software. Estos se ocupan de los problemas recurrentes y son una abstracción del Problema-Solución. Existen patrones de arquitectura, de análisis y de diseño. Un arquitecto puede reutilizar patrones para definir el comportamiento externo e interno de un sistema, así como su estructura. En el año 1977, Christopher Alexander[18] expresa: "*Cada patrón describe un problema que ocurre una y otra vez en nuestro ambiente, y luego describe el núcleo de la solución a ese problema, de tal manera que puedes usar esa solución un millón de veces más, sin hacer jamás la misma cosa dos veces".* ***(12)***

Christopher Alexander 1977 también expresó: *"El patrón es, en suma, al mismo tiempo una cosa que pasa en el mundo y la regla que nos dice cómo crear esa cosa y cuándo se debe crear. Es tanto un proceso como una cosa; tanto una descripción de una cosa que está viva como una descripción del proceso que generará esa cosa."* Estas definiciones dadas por este autor llevan más de 30 años y sin embargo no han perdido su significado en el trayecto de los años, ya que se mantienen en la actualidad.

Mediante los patrones se pueden describir las mejores prácticas, buenos diseños, y encapsular la experiencia de manera tal que otros los puedan reutilizar. Establecen mecanismos donde su objetivo es la solución de problemas que suceden periódicamente en un contexto bien definido.

Los patrones brindan soluciones ya probadas y documentadas a problemas que están sujetos a contextos análogos en el desarrollo de software. Cada patrón permite que algunos aspectos de la estructura del sistema puedan cambiar independientemente de otros aspectos. Además posibilitan la reusabilidad, flexibilidad y el mantenimiento. Dichos patrones se clasifican según el propósito para el que han sido definidos en: **(14)**

- Creacionales: solucionan problemas de creación de objetos, ayudan a encapsular y abstraer dicha creación;
- Estructurales: solucionan problemas de composición y/o agregación de clases y objetos;
- Comportamiento: brindan soluciones respecto a la interacción y responsabilidades entre clases y objetos, así como los algoritmos que encapsulan.

[18] *Es un arquitecto, reconocido por sus diseños destacados en California, Japón y México.*

Existen varias estrategias, arquitecturas y patrones de diseño para el manejo de la lógica de negocio y el acceso a la base de datos. Determinar el uso de una u otra arquitectura está dado por las características del software a desarrollar y también por el gusto de quien diseña el esqueleto de la aplicación. Un patrón que permite el acceso a la bases de datos es el active record.

1.5.1 Patrón de Acceso a Datos: Active Record[19]

Active Record es un patrón en el cual, el objeto contiene los datos que representan a un renglón de nuestra tabla o vista, además de encapsular la lógica necesaria para acceder a la base de datos. De esta forma el acceso a datos se presenta de manera uniforme a través de la aplicación.

Una clase Active Record consiste en el conjunto de propiedades que representa las columnas de la tabla, más los típicos métodos de acceso como las operaciones CRUD[20], búsqueda, validaciones, y métodos de negocio.

Es conveniente utilizar este patrón cuando la lógica del negocio es simple y presenta poca relación con otras entidades; mas si es compleja, el mismo pierde coherencia. Por otro lado es factible utilizarlo cuando la estructura de la tabla coincide con la estructura de la clase pero esta coincidencia implica que si se realiza un cambio en el diseño de la tabla se tiene que cambiar la clase.

1.6 ARQUITECTURA DE DATOS

La arquitectura de datos identifica y define las mejores clases de datos que apoyan las funciones del negocio definidas en el modelo de negocios. Es una de las primeras arquitecturas a ser definidas porque la calidad de los datos es el producto básico. La misma consta de entidades de datos, cada una de las cuales tiene atributos y relaciones con otras entidades de datos; establece las directrices comunes para las operaciones de datos que permitan predecir y controlar el flujo de datos en el sistema; describe los grupos de datos y sus elementos así como asigna los artefactos de datos. Se divide en tres niveles generales: interno, conceptual y externo.

Nivel Interno: es el más cercano al almacenamiento físico, es decir, le concierne la manera de cómo los datos se almacenan en realidad. La arquitectura de datos proporciona criterios para los tratamientos de

[19] *Patrón que se utiliza en la base de datos del GESPRO.*
[20] *Es el acrónimo de Crear, Obtener, Actualizar y Borrar (del original en inglés: Create, Read, Update and Delete).*

los datos que hacen posible el diseño del flujo de datos y además permite controlar el flujo de datos en el sistema.

Nivel Externo: es el más cercano a los usuarios pues se ocupa de la forma en cómo cada usuario ve los datos.

Nivel Conceptual: es un nivel de mediación entre los niveles anteriormente mencionados.

La arquitectura de datos es un conjunto de capas de los modelos que proporciona una base sólida para las iniciativas estratégicas, tales como:

- Una estrategia de datos, destacando los objetivos de la empresa y los objetivos de mejorar la recopilación y uso de datos.
- Mejoras en los procesos de negocios.
- Las decisiones sobre el futuro de los sistemas nuevos y modificados.
- Integración de almacenamiento de datos,
- Las iniciativas de presentación de informes.

El estudio de este tipo de arquitectura resulta de gran importancia para la presente investigación pues permite conocer características elementales para el avance de este trabajo.

1.7 ESCENARIOS EN LA ARQUITECTURA

Los escenarios en la arquitectura ayudan a desglosar las vistas arquitectónicas en partes más fáciles para su correcta definición y elaboración. Estos a su vez hacen menos engorroso el trabajo ya que al estar desglosado se pueden ver las diferentes funcionalidades por separado y así poder centrarse en cada una de ellas sin pensar en la otra.

Los escenarios se describen mediante texto común y a veces se describen mediante dibujos, como por ejemplo diagramas de interacción de objeto. Se acostumbra utilizar UML (en el contexto de 4+1) no tanto como recurso de modelado que después generará alguna clase de código, sino como instrumento de dibujo más o menos informal. Los escenarios constituyen una herramienta importante para relacionar

vistas arquitectónicas, porque recorriendo un escenario se pueden mostrar las formas en que fragmentos o escenas de esas vistas se corresponden entre sí. **(15)**

En la vista de datos., de manera general, los escenarios permiten describir los pasos que se necesitan desarrollar dicha vista, como son las conexiones, los mecanismos de salvas y el tratamiento de los datos en la base de datos.

1.8 DICCIONARIO DE DATOS

El diccionario de datos es un listado organizado de todos los datos que pertenecen a un sistema. Es quien tiene el control de los datos que en el sistema se manejan, controlando las ambigüedades. Estos diccionarios se desarrollan durante la definición de la arquitectura del sistema y su contenido se emplea durante el diseño del proyecto. En un diccionario de datos se encuentra la lista de todos los elementos que forman parte de la base de datos de todo el sistema.

Un diccionario de datos es un conjunto de metadatos que contiene las características lógicas de los datos que se van a utilizar en el sistema que se programa, incluyendo nombre, descripción, alias, contenido y organización. Estos diccionarios se desarrollan durante el análisis de flujo de datos y facilitan el trabajo a los analistas que participan en la determinación de los requerimientos del sistema, su contenido también se emplea durante el diseño del proyecto. **(16)**

En un diccionario de datos se plasman todos los términos de la base de datos de un sistema, en él se describe el significado de cada campo para evitar así las ambigüedades a la hora de desarrollar el sistema y que este no presente conflictos en los términos de la base de datos.

1.9 MODELO DE DATOS

Un **modelo de datos** es un lenguaje que permite describir en una base de datos: **(39)**

- Las estructuras de datos de la base (el tipo de los datos que hay en la base y la forma en que se relacionan)
- Las restricciones de integridad (un conjunto de condiciones que deben cumplir los datos para reflejar correctamente la realidad deseada).
- Las operaciones de manipulación de los datos (operaciones de agregado, borrado, modificación y recuperación de los datos de la base).

1.9.1 Ventajas

- Proveen facilidades para la manipulación de grandes volúmenes de datos.
- Simplifican la programación de equipos de consistencia.
- Garantizan que los cambios de la base sean siempre consistentes sin importar si hay errores con el manejo de las políticas de respaldo adecuadas.
- Organizan los datos con un impacto mínimo en el código de los programas.
- Disminuyen drásticamente los tiempos de desarrollo y aumentan la calidad del sistema desarrollado si son bien explotados por los desarrolladores.
- Proveen interfaces y lenguajes de consulta que simplifican la recuperación de los datos.

1.9.2 Desventajas

- Si se tienen muy pocos datos que son usados por un único usuario por vez y no hay que realizar consultas complejas sobre los datos, entonces es posible que sea mejor usar una hoja de cálculo.
- Si el software es muy complejo, las personas que vayan a utilizar el modelo de datos, deben tener conocimiento de las funcionalidades del mismo para poder aprovecharlo al máximo.
- La complejidad y la gran cantidad de funciones que tienen hacen que sea un software de gran tamaño, por lo que requiere de gran cantidad de memoria para ejecutarse.

1.10 HERRAMINETAS CASE PARA EL MODELADO DE BASES DE DATOS

Estas herramientas posibilitan el diseño de un software antes de su codificación, esto se realiza mediante diagramas que permiten que todo el equipo de desarrollo comprenda lo que se quiere hacer realmente y que el trabajo que se realice sea de forma homogénea. Algunas de estas herramientas tienen un valor económico muy alto y requieren altos costos de entrenamiento para el personal. Por otra parte, algunas herramientas CASE[21] no ofrecen o evalúan soluciones potenciales para los problemas relacionados con sistemas, o simplemente no llevan a cabo ningún análisis de los requerimientos de la aplicación.

[21] *Computer Aided Software Engineering, Ingeniería de Software Asistida por Computadora son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y de dinero.*

Entre las principales herramientas se encuentran:

- ***Embarcadero ER / Studio:*** Es una herramienta de modelado de datos fácil de usar para el diseño y construcción de bases de datos a nivel físico y lógico. Direcciona las necesidades diarias de los administradores, desarrolladores y arquitectos que construyen y mantienen aplicaciones de bases de datos grandes y complejas. Soporta las bases de datos: Oracle, Microsoft Access, Microsoft SQL Server, entre otras.

- ***Visual Paradigm:*** Es una herramienta CASE que utiliza UML como lenguaje de modelado, soporta el diagrama entidad-relación. Está disponible en varias ediciones[22] que permite escoger la más idónea para el usuario según su necesidad. Esta herramienta se distribuye bajo licencia gratuita y comercial. Entre las principales características que presenta Visual Paradigm están: producto de calidad, multiplataforma, soporta aplicaciones web, varios idiomas, fácil de instalar y actualizar, compatibilidad entre ediciones, soporta a miles de usuarios trabajando sobre el mismo proyecto y permite que cada uno vea los cambios realizados en tiempo real.

A partir de un análisis realizado a las principales herramientas CASE anteriormente expuestas se concluyó que para el modelado de la base de datos de GESPRO se utilizará Visual Paradigm para UML. Este software brinda múltiples ventajas al usuario como son: es fácil de instalar y actualizar, soporta el PostgreSQL y es de libre distribución, además es un software multiplataforma. Visual Paradigm permite realizar todo tipo de diagramas de clases, generar código desde diagramas y generar documentación. Además facilita la generación de bases de datos, transformación de diagramas de Entidad-Relación en tablas de base de datos e ingeniería inversa.

1.11 SISTEMAS DE GESTIÓN DE BASES DE DATOS

En la actualidad se han desarrollado un conjunto de gestores de bases de datos. Entre los más destacados en la industria del software se encuentran:**(Figura 3) (40)**

- PostgreSQL
- MySQL
- Microsoft SQL Server

[22] Enterprise, Professional, Community, Standard, Modeler y Personal.

- Oracle

A continuación se muestra una tabla que establece una comparación entre los aspectos fundamentales que se deben tener en cuenta para la selección de alguno de ellos.

Criterios de comparación	Oracle	MySQL	SQL Server	PostgreSQL
Sistemas Operativos	Windows, Mac OS X, Linux y GNU/UNIX	Windows, Mac OS X, Linux, BSD y GNU/UNIX	Windows	Windows, Mac OS X, GNU/Linux, BSD y UNIX
Consumo de recursos	Necesita gran cantidad de recursos	Consume pocos recursos	Consume pocos recursos.	Consume pocos recursos
Tipo de licenciamiento	Propietario	GPL o Propietario	Propietario	BSD
Integridad referencial	Si	No	Si	Si

Figura 3. Comparación entre los diferentes gestores de base de datos

1.11.1 PostgreSQL

Siguiendo la política de la Universidad de las Ciencias Informáticas de hacer uso de herramientas de desarrollo que sean libres, se decide utilizar como gestor de base de datos PostgreSQL. Este se encuentra catalogado como la mejor y más avanzada base de datos Open Source[23] del mundo.

Iniciado como un proyecto universitario en la década de 1980, ha llevado varios períodos de desarrollo y cuenta con una arquitectura confiable, estabilidad de procesamiento e integridad en el manejo de datos. Es un verdadero proyecto de software abierto, ya que su desarrollo no es dirigido por una empresa sino que es administrado por una comunidad.

PostgreSQL es un sistema de bases de datos relacional orientado a objetos, ya que incluye características como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema **(Figura 4)**. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.

[23] *Base de datos de código abierto.*

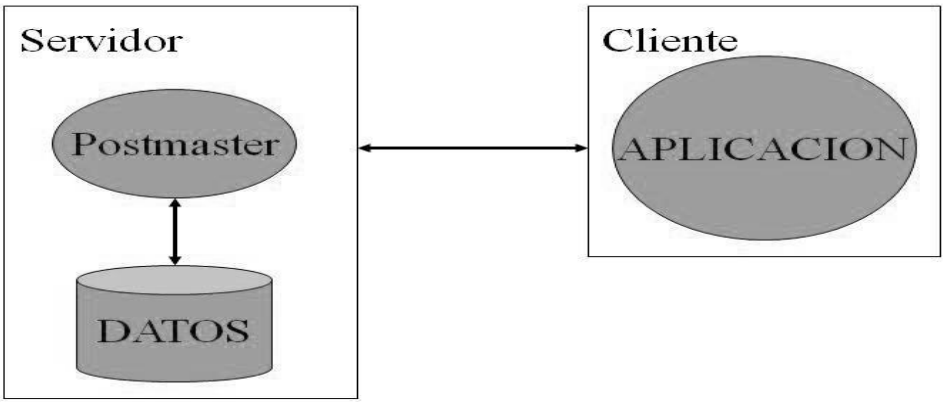

Figura 4. Modelo Cliente-Servidor de PostgreSQL

Características de PostgreSQL:

- Utiliza el Control de Concurrencia Multi-Versión, por sus siglas en inglés MVCC (Multi-Version Concurrency Control), como tecnología para evitar bloqueos innecesarios.
- Es multiplataforma.
- Soporta claves foráneas.
- Posee buenas interfaces de instalación y administración.
- Hace uso de Write Ahead Logging (Escritura anticipada del registro): incrementando la dependencia de la base de datos al registro de cambios antes de que estos sean escritos en la base de datos. Esto garantiza que en el hipotético caso de que la base de datos se caiga, existirá un registro de las transacciones a partir del cual se podrá restaurar la base de datos.
- Cumple con las propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad).
- Ha sido diseñado para mantenerse estable con grandes volúmenes de datos y transacciones.
- Es altamente configurable, con lo cual puede personalizarse de acuerdo al escenario donde será utilizado.

Para la administración y desarrollo con PostgreSQL se utilizará la herramienta pgAdmin III[24]. Esta es una aplicación gráfica para gestionar PostgreSQL, siendo la más completa y popular con licencia Open Source. Está escrita en C++ usando la librería gráfica multiplataforma wxWidgets, lo que permite que se

[24] *Herramienta para la administración de bases de datos.*

pueda usar en Linux, FreeBSD, Mac OS X y Windows[25]. Es capaz de gestionar versiones a partir de la de PostgreSQL 7.3 ejecutándose en cualquier plataforma.

Esta herramienta de administración de bases de datos está diseñada para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y facilita enormemente la administración. La conexión al servidor puede hacerse mediante el protocolo TCP/IP.

CONCLUSIONES DEL CAPÍTULO

Para guiar el desarrollo de la arquitectura de software de la vista de datos del GESPRO 12.05, se decidió hacer uso del modelo de las 9+1 vistas que propone la Universidad de las Ciencias informáticas, ya que este posee una vista dedicada exclusivamente al tratamiento de los datos de un sistema en la cual se definen 4 pasos fundamentales que guían el diseño de la arquitectura de esta vista.

Para el modelado de la base de datos se escogió el Visual Paradigm como herramienta de modelado ya que es una herramienta libre y sus ventajas descritas facilitan el trabajo.

Por otro lado se propone que la herramienta gestora de la base de datos sea PostgreSQL en su versión 8.0 o una superior a esta porque además de ser libre es multiplataforma, necesita pocos recursos para su funcionamiento y es un gestor muy fácil de aprender.

[25] *Sistemas operativos.*

Capítulo 2. Diseño Arquitectónico de la vista de datos del GESPRO 12.05

En este capítulo se trata una panorámica general del diseño de la arquitectura del sistema de GESPRO 12.05. Se analiza y se caracteriza el producto según la vista, dando una definición de la misma. Se desglosan los diferentes escenarios definiendo cuáles de ellos son los que rigen la implantación de la vista de datos en el sistema de GESPRO y se concreta un patrón de solución para cada escenario. Se construye un diccionario de datos para evitar conflictos a la hora de definir los términos en las bases de datos, y se diseña tanto el modelo físico como lógico de la base de datos.

2.1 CARACTERIZACIÓN DEL GESPRO SEGÚN LA VISTA DE DATOS

La vista de datos del sistema GESPRO en su versión 12.05 presenta diferentes características esenciales que sustenta la arquitectura de la vista.

En la base de datos (BD) del sistema de GESPRO de la Universidad de las Ciencias Informáticas (UCI) se almacenan información de tipo: string, integer, double precisión, text, character varying[26]; ya que en los proyectos de la Universidad la información que se maneja principalmente son datos.

La información que se guarda en la base de datos del GESPRO es gestionada dentro de la misma herramienta de gestión, o sea, que los mismo usuarios del sistema son los que introducen esta información a la BD. Por tal razón no se hace necesario la consulta y almacenamiento de alguna fuente externa.

En la base de datos del sistema de gestión de la universidad se almacena y va destinada hacia ella toda la información que se necesita y manipulan en los proyectos productivos, esta información se guarda de manera temporal mientras se haga necesaria la consulta y manipulación de los datos de los proyectos, por un período de un mes. Pasado el mes, se elimina la información según la disponibilidad de la base de datos.

A pesar de la UCI contar con 17 centros productivos y cada uno con diferentes proyectos, la información guardada en la BD del sistema de GESPRO de la Universidad no es de gran volumen, cada centro

[26] *Tipos de datos que se almacenan en la base de datos de GESPRO.*

productivo consta de una instancia de la base de datos, donde se guarda la información de cada uno por separado. Por tal razón es que esta BD presenta un volumen de información que no es mayor a 1 GB[27].

De manera resumida las características de la vista de datos de GESPRO son:

1. En las bases de datos se almacena información de tipo datos.
2. La fuente de datos e s de origen interno.
3. El destino de las fuentes de información es de almacenamiento y procesamiento temporal.
4. El volumen de la información que se desea almacenar es medio.

La vista de la arquitectura de datos posea una descripción de cada uno de los escenarios que intervienen en su funcionamiento, además, cuenta con una definición del diccionario de la base de datos y con el modelo relacional de la misma. A continuación se define un conjunto de soluciones que se deben de tener en cuenta para el correcto funcionamiento de esta vista de la arquitectura.

2.2 DESCRIPCIÓN DE LOS ESCENARIOS Y PATRONES DE SOLUCIÓN EN LA VISTA DE LA ARQUITECTURA DE DATOS DEL GESPRO 12.05

La vista de datos de sistema de gestión de la Universidad de las Ciencias Informáticas (GESPRO) presenta diferentes escenarios, los cuales hacen que esta sea más eficiente a la hora de realizar su implementación en el sistema GESPRO 12.05.

Cada escenario presenta cuatro partes fundamentales. Ellas son:

1. Descripción breve del escenario.
2. Los diferentes patrones de solución que permiten el buen funcionamiento de la vista.
3. Las restricciones que trae consigo aplicar estos patrones.
4. Los atributos de calidad que se mejoran aplicando la solución propuesta.

2.2.1 Manejo del Pool de conexiones

[27] *Se refiere a las siglas de la unidad de almacenamiento GigaBit.*

Descripción: En el sistema de gestión (GESPRO) este escenario es el encargado de manejar las conexiones a las bases de datos posibilitando un trabajo más eficiente del gestor de datos y de la aplicación en general.

Patrón de Solución: La arquitectura tecnológica de la plataforma de GESPRO, proveerá algún mecanismo o componente de administración y configuración del Pool de conexiones[28] con los gestores más utilizados (PostgreSQL, Oracle, etc.). De esta manera los componentes de la base de datos del sistema de gestión cuentan con una interfaz única de configuración y el marco de trabajo controla centralizadamente todas las conexiones y podrá auditar y tomar acciones que mejoren el rendimiento y la trasmisión de datos. Como patrón de solución para este escenario se plantea que la variante a utilizar en el GESPRO debe ser la conexión por usuarios, dónde se construyen conexiones por cada usuario y se mantienen durante el tiempo de vida del usuario en la aplicación. Esto ocurrirá con tantas conexiones como sean necesarias, con tiempos de vida limitados en función de diferentes contextos, posibilitando que no existan conexiones por largos períodos de tiempo.

Restricción: Este patrón de solución tiene como restricción que en el sistema de gestión se debe mantener un adecuado balance entre la estrategia de conexiones por contexto contra otras variantes y esto dependerá del tipo de aplicación y de la naturaleza de los datos y operaciones que se ejecuten.

Atributo de calidad: Con la aplicación de este patrón de solución para este escenario, se pretende lograr una mejor eficiencia de la base de datos del sistema de gestión de proyecto en la Universidad de las Ciencias Informáticas, con vistas a que la base de datos no presente errores en el manejo de las conexiones.

2.2.2 Transparencia entre la capa de negocio y la capa de datos

Descripción: Este escenario en el GESPRO requiere flexibilidad de la aplicación de forma tal que no haya grandes afectaciones en caso de ser necesario realizar modificaciones en las bases de datos.

Patrón de Solución: Para resolver la situación en el escenario descrito para el sistema de gestión, se plantean los siguientes aspectos a tener en consideración:

[28] *(Agrupamiento de conexiones) es el manejo de una colección de conexiones abiertas a una base de datos de manera que puedan ser reutilizadas al realizar múltiples consultas o actualizaciones.*

- La arquitectura tecnológica de la plataforma proveerá algún mecanismo de abstracción de la capa relacional de persistencia. Este mecanismo de mapeo relacional de objeto (por sus siglas en inglés ORM), permitirá contar con algún mecanismo de SQL Helper[29].

- EL ORM debe permitir en la base de datos del sistema una persistencia compuesta, una carga perezosa, una actualización y modificación sincronizada; tanto en el modelo mapeado como en el modelo relacional, con el fin de contar con los pasos necesarios para lograr la sincronización práctica y eficiente.

- El ORM cuenta con administración de conexiones y de transacciones, y con un lenguaje script SQL o abstracción de este, de manera tal que se puedan realizar operaciones de consulta u operaciones CRUD[30] sobre el esquema mapeado; ejecutándose los mismos en la base de datos de forma transparente para el desarrollador o usuario de la plataforma.

Esta solución que se plantea es aplicable a la solución del escenario para lograr una correcta función de la base de datos del sistema de gestión.

Restricción: Este patrón de solución tiene como limitante que al aplicarse puede afectar el rendimiento de la aplicación.

Atributo de calidad: Se pretende que la base de datos mantenga una mantenibilidad que no afecte el rendimiento de la herramienta.

2.2.3 Administración de transacciones

Descripción: Con frecuencia en los sistemas de información se producen transacciones que por su tamaño pueden verse afectadas por causas internas o externas a la aplicación informática. Las soluciones informáticas deben proveer facilidades para la correcta ejecución de las transacciones. Así mismo las operaciones de modificación de la información podrían ser consideradas como transaccionales facilitando la gestión de la información y la consistencia de la misma. El sistema de gestión de proyecto de la Universidad de las Ciencias Informáticas no es la excepción de lo planteado anteriormente, y para

[29] *Es una clase que proporciona un conjunto de facilidades, que permiten ejecutar varios tipos de co- mandos con la base de datos.*
[30] *Es el acrónimo de Crear, Obtener, Actualizar y Borrar (del original en inglés: Create, Read, Update and Delete).*

mantener una correcta transacción en la base de datos[31] de GESPRO se plantea la siguiente solución para el escenario descrito.

Patrón de Solución: Se plantea que la arquitectura tecnológica de la plataforma de GESPRO provee algún mecanismo de administración de transacciones, de manera que las operaciones de modificación sobre el modelo de dominio sean transaccionales por defecto. Para esto se deben cumplir los siguientes aspectos:

- Permitir que el mecanismo sea transparente para el programador.
- Facilitar que la solución arquitectónica configure el esquema transaccional de un dominio específico de la solución que se construye, definiéndose en este caso para una transacción de negocio, cuándo inicia y cuándo termina.
- Permitir configurar notificaciones ante la posibilidad de fallas en la transacción por un mecanismo central de notificaciones, que implemente para toda la tecnología un administrador de patrón observer[32], o podrá ser inyectadas por algún mecanismo de inversión de control.
- Capturar como un evento más el rollback[33] de la transacción.

Restricción: Si no es manejado adecuadamente puede afectar el rendimiento de las soluciones en caso de que se implementen de forma transaccional procesos muy largos que no necesariamente requieren este tipo de tratamiento.

Atributo de calidad: Pretende que la funcionalidad del sistema y la fiabilidad de la información guardada en la base de datos mejoren considerablemente con respecto a la versión anterior del sistema de gestión.

2.2.4 Concurrencia en el acceso a datos

Descripción: En cuanto a la concurrencia en el acceso a datos en la base de datos del GESPRO se tiene que un mismo campo puede ser accedido por uno o varios clientes al unísono con el fin de realizar determinada acción; en especial a la hora de realizar una modificación. Con este escenario se va a dar

[31] Es un conjunto de órdenes que se ejecutan formando una unidad de trabajo y estas son capaces de mantener la integridad de los datos en un sistema.
[32] Es un patrón de diseño que define una dependencia del tipo uno-a-muchos entre objetos, de manera que cuando uno de los objetos cambia su estado, notifica este cambio a todos los dependientes
[33] En tecnologías de base de datos, es una operación que devuelve a la base de datos a algún estado previo. Los Rollbacks son importantes para la integridad de la base de datos, a causa de que significan que la base de datos puede ser restaurada a una copia limpia incluso después de que se han realizado operaciones erróneas.

una variante de solución para eliminar el problema en cuanto a la concurrencia y demora a la hora de acceder a la base de datos.

Patrón de solución: Para eliminar el problema en cuanto a la concurrencia y demora a la hora de acceder a la base de datos del GESPRO es necesario proveer algún mecanismo de configuración y gestión dinámica[34] de las características de concurrencia sobre entidades o dominios de la solución lógica que se modela, de manera que se pueda configurar el esquema de concurrencia que se desee sobre las entidades (conceptos) o estructuras de entidades del esquema de persistencia de una solución específica. Las opciones de concurrencia deberán permitir implementar el mecanismo pesimista[35].

Este mecanismo es el encargado de realizar el control desde la base de datos. A cada tabla (no a los nomencladores) se le agregaría un campo versión que permita el control de las modificaciones. Para eso es necesario tener en cuenta los siguientes aspectos en la base de datos del GESPRO:

- El acceso a los datos puede estar afectado por alguno de los siguientes estados: lectura, escritura o bloqueado.
- Las características de instanciación[36] en el esquema de concurrencia deben estar asociadas al valor de algún atributo del concepto o a rol de acceso específico.
- Todo el tiempo se chequea por este campo el acceso a datos, durante la actualización y consulta de los mismos.

Restricción: Esta solución puede afectar el rendimiento del sistema de gestión de proyecto de la Universidad de las Ciencias Informáticas (UCI).

Atributo de calidad: Con esta solución se pretende que la confiabilidad de los datos que se almacenan en el sistema sea la más óptima posible, para que no se pierda la información y esta sea lo más precisa posible.

---

[34] Se refiere a gestionar los datos en la base de datos de manera rápida y ágil.

[35] El punto de vista pesimista es el que considera que muchas transacciones tienen conflictos con otras. Los algoritmos pesimistas sincronizan la ejecución concurrente de las transacciones en la etapa inicial de su ciclo de ejecución.

[36] La palabra Instancia significa: Solicitud o Insistencia. Una instancia de un programa es una copia de una versión ejecutable del programa, que ha sido escrito en la memoria del computador.

2.2.5 Tratamiento con estructura de árboles desde la base de datos

Descripción: En el sistema de gestión de la UCI el escenario encargado del tratamiento con estructuras de árboles desde la base de datos, se basa en que los datos jerárquicos son una colección de datos donde cada artículo tiene un solo padre, puede tener muchos o ningún hijo (con la excepción del nodo raíz, este no tiene ningún padre). Estos datos pueden tener múltiples aplicaciones en la base de datos del GESPRO y la recuperación de los mismos son elementos fundamentales para el sistema.

Patrón de solución: El modelado de una estructura de árbol en la base de datos del GESPRO permite el conocimiento de todos los hijos pertenecientes a un determinado padre y viceversa. Para esto se cuenta con cuatro atributos: Id, Id padre, ordenizq[37], ordender[38]**.** La búsqueda funcional es contraria al modelo de lista de adyacencia, pues esta búsqueda funciona independiente de la propiedad del árbol, no hay que preocuparse con el valor RGT del nodo, pues esto se soluciona con la cláusula BETWEEN[39], porque el valor siempre caerá dentro del mismo progenitor así como los valores LTF. (Árboles)

Restricción: Esta solución puede afectar la eficiencia por las frecuentes consultas a la base de datos y el acceso a ficheros sino se trabaja con cuidado el tratamiento de la caché y de las recuperaciones de la información.

Atributo de calidad: Si se aplica correctamente este patrón se mejora considerablemente la eficiencia del sistema y con él el rendimiento de la herramienta.

2.2.6 Réplica de datos

Descripción: La réplica de datos en la base de datos del sistema de gestión de proyectos de la UCI requiere de un entorno dónde se transfieran datos de una base de datos para otra.

Patrón de solución: La arquitectura tecnológica de la plataforma de GESPRO provee algún mecanismo que permita replicar, intercambiar y transformar información entre diferentes bases de datos, iguales o diferentes en estructura y gestores de bases de datos, en línea y fuera de línea (con o sin conexión), unidireccional o bidireccionalmente. Se utilizarán algunas de las herramientas relacionadas en la vista de entorno de desarrollo tecnológico.

---

[37] *Se refiere a orden izquierdo.*
[38] *Se refiere a orden derecho.*
[39] *Permite la selección de un rango determinado.*

Restricción: Se debe trabajar adecuadamente en la solución de réplica teniendo en cuenta los requerimientos de los clientes para evitar problemas en el sistema de gestión de la UCI.

Atributo de calidad: La funcionalidad del sistema crece considerablemente si se aplica este patrón de solución para el escenario descrito en la vista de la arquitectura de datos del GESPRO.

2.2.7 Procesamiento de los datos

Descripción: En el sistema de gestión de proyecto de la Universidad de las Ciencias Informáticas el procesamiento de los datos se desarrolla de diferentes maneras. Para ello se aplican dos algoritmos que permiten el tratamiento de los datos en la base de datos del GESPRO.

Patrón de solución: Un algoritmo aplicado en el tratamiento de los datos es el de las técnicas estadísticas para la minería de datos. Para aplicar este algoritmo es necesario aplicar ciertas técnicas, las cuales se muestran a continuación:

1. Limpieza de datos.
2. Identificación y sustitución por sinónimos estandarizando los datos.
3. Estandarización de términos.
4. Sustitución de datos ausentes con algunas de las siguientes técnicas (*imputación*).
5. Detección y estandarización de datos fuera de rango.
6. Construcción asistida de tareas para eliminar posibles fuentes de error e inconsistencias (cronogramas tipo, generación asistida de tareas a partir de requisitos).
7. Identificación de problemas de edición y resolución con técnica de Comparación de Distancias de Edición.

Otro algoritmo que se aplica en la base de datos de GESPRO para el tratamiento de los datos en el sistema, es el de las técnicas de inteligencia artificial para lo cual es necesario aplicar las siguientes técnicas:

1. Cálculo de los indicadores (Inteligencia artificial).
2. Sistemas de inferencia borrosa[40] de tipo Mamdani[41].

[40] *Un sistema de inferencia borrosa (FIS, Fuzzy Inference System en inglés) es una forma de transformar un espacio de entrada en un espacio de salida.*

3. Sugeno Grado Cero (Evaluación del proyecto)[42].

Estas técnicas o algoritmos son las que permiten el mantenimiento y tratamiento de los datos en la base de datos del sistema de GESPRO. Con ellas se posibilita que la base de datos no se cargue y se le den tratamientos a los datos que ya no son necesarios en los proyectos por cuestiones de mantener actualizada la base de datos.

Restricción: La selección inadecuada de estos algoritmos puede provocar que no se obtengan los resultados esperados y además afectar significativamente el rendimiento de la solución.

Atributo de calidad: La funcionalidad de la herramienta de gestión y la eficiencia de la misma se espera que mejoren en consideración con la versión 11.05 del sistema si se aplica este patrón de solución a la base de datos del GESPRO.

2.2.8 Configuración básica del servidor de datos

Descripción: La configuración básica del servidor de datos en el sistema de gestión de proyectos de la Universidad de las Ciencias Informáticas (UCI) debe ser configurado garantizando un rendimiento adecuado de la base de datos.

Patrón de solución: Es preciso realizar varias acciones para la configuración de las variables de rendimiento del servidor. Entre las acciones que se pueden encontrar están:

- Destinar un equipo a la administración y tunning[43] de las bases de datos.
- Documentar procedimiento para la configuración y ajuste de los parámetros que influyen en el rendimiento del gestor de bases de datos, especificando claramente los parámetros adecuados para las siguientes variables.
    a) Cantidad de conexiones concurrentes
    b) Consumo de memoria por sesión

---

[41] El método de Mamdani es un método para la inferencia borrosa, es el más usado en aplicaciones, dado que tiene una estructura muy simple de operaciones "mín-max".
[42] Es un método que se aplica para la inferencia borrosa en las bases de datos. Este método es contrario al Mamdani.
[43] Un tunning de base de datos es la adecuación de procesos que se mantienen en memoria para dar acceso a esa base de datos.

c) Limite de tamaño máximo de tablas temporales

d) Tamaño de memoria virtual

e) Tamaño de memoria chache del servidor

Restricción: Si no se dedica un equipo al tema y se planifican adecuadamente las acciones antes mencionadas, no se lograría una configuración del servidor de datos del GESPRO.

Atributo de calidad: Se espera que la funcionalidad, mantenibilidad, eficiencia y fiabilidad de la herramienta mejoren considerablemente con respecto a la versión anterior del sistema de gestión de proyectos del UCI.

2.2.9 Salva y respaldo de las bases de datos

Descripción: Para garantizar una salva y respaldo de los datos del sistema de gestión de la UCI, es necesario montar una infraestructura para garantizar que los datos sean recuperados fácilmente en caso de algún fallo en la base de datos del GESPRO.

Patrón de solución: Entre las acciones a realizar para garantizar un respaldo seguro de las salvas de la base de datos del sistema de gestión de la UCI están:

- Establecer sistema de periodicidad de las salvas frecuentes.

Ejemplo: La estrategia de respaldo comprende la realización de copias de seguridad, con periodicidad diaria, garantizando una copia exacta de la información disponible en el servidor de Base de Datos.

- Establecer sistema de periodicidad de las salvas históricas.

Adicionalmente los datos históricos, en correspondencia con los requerimientos especificados por los especialistas funcionales, se encontrarán activos por 10 años; los datos que superen este período serán respaldados en un área pasiva.

- Realizar estrategias de salva ante amenazas previstas.

Considerando que la oficina podría realizar cambios previstos en la gestión de riesgos se incluye además un respaldo por demanda que puede realizarse en el momento oportuno y se guardaría como una versión superior de los datos ya respaldados anteriormente.

Se incluye un procedimiento para estos casos emergentes.

- Alcance del respaldo

Es recomendable siempre que se pueda y sea necesario hacer backup[44] de toda la base de datos.

- Periodicidad de reemplazo de los Backups

Se hace un backup completo de la base de datos diariamente y se guardan copias de bases de datos anteriores en dependencia con los recursos tecnológicos y las políticas de protección de la información de la organización. Se establece política de reemplazo en consonancia con la cantidad de copias establecidas.

- Pruebas periódicas a los Backups

Las tareas de recuperación y respaldo se realizarán mensuales; para ello se utilizará un servidor pasivo que se alimenta de los respaldos realizados sobre la base de datos de producción. La política asegura un tiempo mínimo de recuperación ante fallas y prueba constantemente las copias realizadas.

Restricción: Si se aplica este patrón de solución se tienen dos restricciones que puedes influir en la correcta aplicación del mismo.

1. La aplicación de este patrón de solución induce a la compra de tecnologías asociadas que pueden implicar un elevado costo.
2. La aplicación de este patrón y los parámetros que en él se reflejan influyen en los procesos de gestión de la información de la organización y deberán estar reflejados en los mismos.

Atributo de calidad: Si se aplica este patrón de solución para el escenario descrito se pretende mejorar la funcionalidad, mantenibilidad y fiabilidad del sistema para lograr un mejor rendimiento del GESPRO.

2.2.10 Política de evolución y crecimiento de los datos

Descripción: En el sistema de gestión de proyectos en la UCI se requiere montar una infraestructura para el control de la evolución y el crecimiento de los datos en función de garantizar elevados niveles de operatividad de la solución y de respuesta del gestor de bases de datos.

[44] Una copia de seguridad o backup (su nombre en inglés) en tecnología de la información o informática es una copia de seguridad, con el fin de que estas copias puedan utilizarse para restaurar el original después de una eventual pérdida de datos.

Patrón de solución: Se plantea que se deben realizar varias acciones para garantizar un buen funcionamiento de la base de datos de GESPRO en cuanto a este escenario. Las acciones que se aplican son las siguientes:

- Realizar la estimación de tamaño de la solución.
- Definir el tiempo de vida útil de los datos en el sistema de información, vista desde la operatividad de los datos y del sistema. La vida útil de los datos en la base de datos es de un mes aproximadamente, estos después del mes se eliminan según la disponibilidad en la base de datos.
- Determinar estrategia para la salva de la información una vez que sea considerada no operativa; así como la recuperación de los datos en caso necesario. Esta acción debe estar en consonancia con la estrategia de salva.

Restricción: La aplicación de este patrón traería las siguientes restricciones en el sistema:

1. La aplicación de este patrón de solución induce a la compra de tecnologías asociadas que pueden implicar un elevado costo.
2. La aplicación de este patrón y los parámetros que en él se reflejan influyen en los procesos de gestión de la información de la organización y deberán estar reflejados en los mismos.

Atributo de calidad: Se pretende que con la correcta aplicación de la solución descrita se mejore la funcionalidad, mantenibilidad y fiabilidad del sistema de gestión de proyecto de la Universidad de las Ciencias Informáticas.

2.2.11 Indexado de la base de datos

Descripción: Para realizar el indexado de la base de datos del sistema de gestión de proyectos de la Universidad de las Ciencias Informáticas se requiere utilizar herramientas CASE para la modelación de las bases de datos, garantizando facilidad, la actualización y modificación del modelo de datos.

Patrón de solución: La solución de este escenario está orientada a los aspectos que se describen en las acciones que se explican a continuación.

1. Estudiar a profundidad cómo el optimizador del RDBMS[45] funciona, o sea, qué tipo de indexado

[45] *Un sistema de bases de datos relacionales (RDBMS) es un sistema de gestión de bases de datos (DBMS) que se basa en el modelo relacional introducido por E. F. Codd.*

utiliza la arquitectura del gestor de bases de datos.

Ejemplo: el gestor PostgreSQL para la búsqueda de datos utiliza las llaves primarias y foráneas. Todas las llaves primarias poseen índices de tipo “b-tree” (Árboles-B) lo que implica que cualquier búsqueda que se realice utilizando las llaves se optimizará mediante este método.

2. A partir de esto, se escoge el método de indexado adecuado ya sea B-tree que hasta ahora es el más usado, The Bitmapped ó The Hash[46].
3. Seleccionar las consultas más frecuentemente utilizadas.
4. Construir los índices necesarios para garantizar elevados niveles de eficiencia en los resultados de las consultas. Mostrar especial interés en las llaves del negocio que se encuentran almacenadas en las tablas y no son representadas como llaves primarias dentro de la base de datos.

Restricción: Al aplicar este patrón de solución se tiene que la utilización de más índices sobre la tabla de hechos u otras estructuras no es una tarea fácil de realizar, debido a que un consumo excesivo de índices lejos de mejorar el rendimiento del sistema atenta de forma directa a su relentización.

Atributo de calidad: La aplicación de este patrón de solución mejora la eficiencia de la base de datos del GESPRO.

2.3 DICCIONARIO DE DATOS DE LA BASE DE DATOS DEL GESPRO

El diccionario de datos del sistema de gestión (GESPRO) de la Universidad de las Ciencias Informáticas está bien definido y conceptualizado de manera correcta para evitar de esta manera las ambigüedades y conflictos que pueden existir en la base de datos a la hora de desarrollarla. La base de datos de GESPRO cuenta con 135 tablas y cada tabla cuenta con diversos campos, los cuales están correctamente definidos para evitar así la mala implantación de la base de datos.

Para el diccionario de datos del GESPRO el lenguaje que se definió es el inglés ya que las mayorías de las bases de datos de las grandes compañías en el mundo utilizan este idioma; existen términos predefinidos por las grandes empresas productoras de software que coinciden, como son: id, name, description, type, text, action, entre otros; es considerado por su alto nivel de empleo en las principales y más prestigiosas bibliografías procedentes de los países desarrollado, como el idioma universal. El

[46] *Métodos de indexado para las bases de datos.*

diccionario cuenta con 350 términos en dependencia de los campos de las tablas, a cada término se le determinó su nombre, descripción, los valores que toma y su significado. A continuación se le muestra unos ejemplos del diccionario de datos realizado.

Ejemplo 1:

Nombre del concepto o variable:

agreement_statuses

Descripción de la variable.

Nombre de la tabla de estados de los acuerdos.

Valores que toma la variable:

Valores que toma	Significado intrínseco
En sus campos se guardan diferentes tipos de datos	Datos relacionas con la tabla en cuestión

Ejemplo 2:

Nombre del concepto o variable:

agreement

Descripción de la variable.

Hace referencia a la tabla de los acuerdos

Valores que toma la variable:

Valores que Toma	Significado intrínseco
En sus campos se guardan diferentes tipos de datos	Datos relacionas con la tabla en cuestión

Ejemplo 3:

Nombre del concepto o variable:

id

Descripción de la variable.

Hace referencia al indicador de una tabla

Valores que toma la variable:

Valores que Toma	Significado intrínseco
Serial	Toma valores numéricos.

Ejemplo 4:

Nombre del concepto o variable:

project_id

Descripción de la variable.

Hace referencia al identificador de la tabla project

Valores que toma la variable:

Valores que Toma	Significado intrínseco
Enteros	Toma los valores que toma el identificador en la tabla que se relaciona

2.4 MODELO DE DATOS DE LA BASE DE DATOS DEL GESPRO

El modelo de datos del sistema de gestión de proyectos (GESPRO) está compuesto por 135 tablas. Estas están en correspondencia con la necesidad de la universidad de gestionar los proyectos. Entre ellas se destinan 65 a todo lo relacionado con los proyectos de la universidad, ya se diga cambio en los proyectos, requisitos de los proyectos, requerimientos, estado, errores, entre otras funcionalidades. Se destinan 10

tablas a los procesos, 18 a las peticiones, y las demás están destinadas a otras funcionalidades necesarias en el sistema.

La base de datos de GESPRO cuenta con tablas importantes para la gestión de los proyectos, estas son las vinculadas a los proyectos, las peticiones, los usuarios, los custom fields y custom values. A continuación se muestran algunas de las relaciones que presentan estas tablas en la base de datos.

Están las que pertenecen a la administración de los proyectos, o sea la que están vinculas a crear proyectos y todo lo relacionado con los mismos **(Figura 5).** Estas tablas son de fundamental importancia en el sistema de GESPRO, ellas se encargan de mantener guardada toda la información perteneciente a un proyecto determinado, y así permitir la gestión de los datos de cada proyecto.

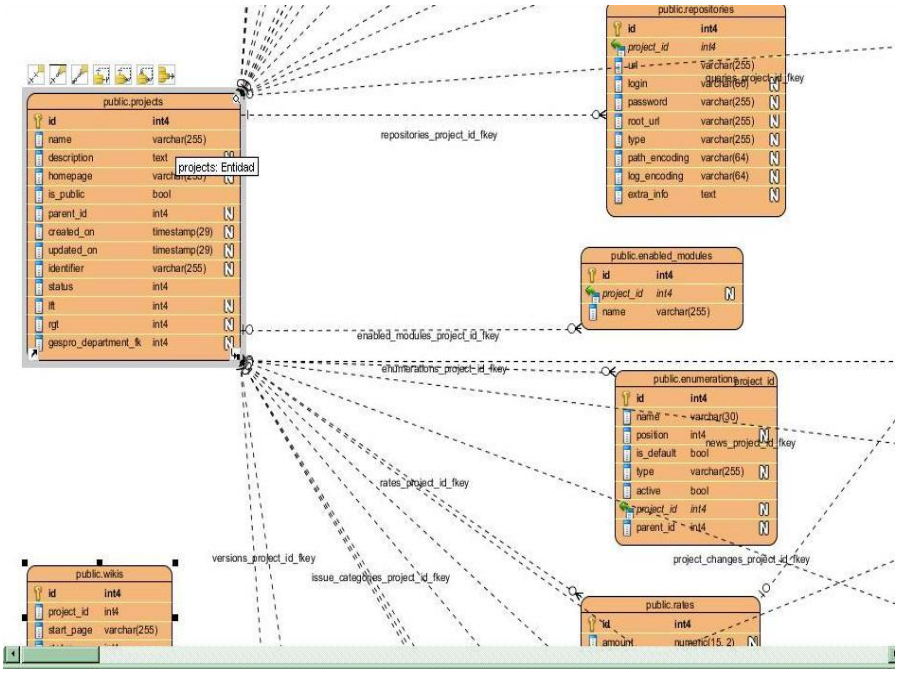

Figura 5. Relaciones de las tablas proyecto

La tabla user (usuario) también es de gran importancia en la base de datos de GESPRO, los usuarios son los encargados de gestionar los proyectos en la aplicación, de introducir, eliminar, modificar y consultar los datos de cada proyecto. Esta tabla está relacionada al igual que la tabla proyectos, con la mayoría de las tablas de la base de datos **(Figura 6)**.

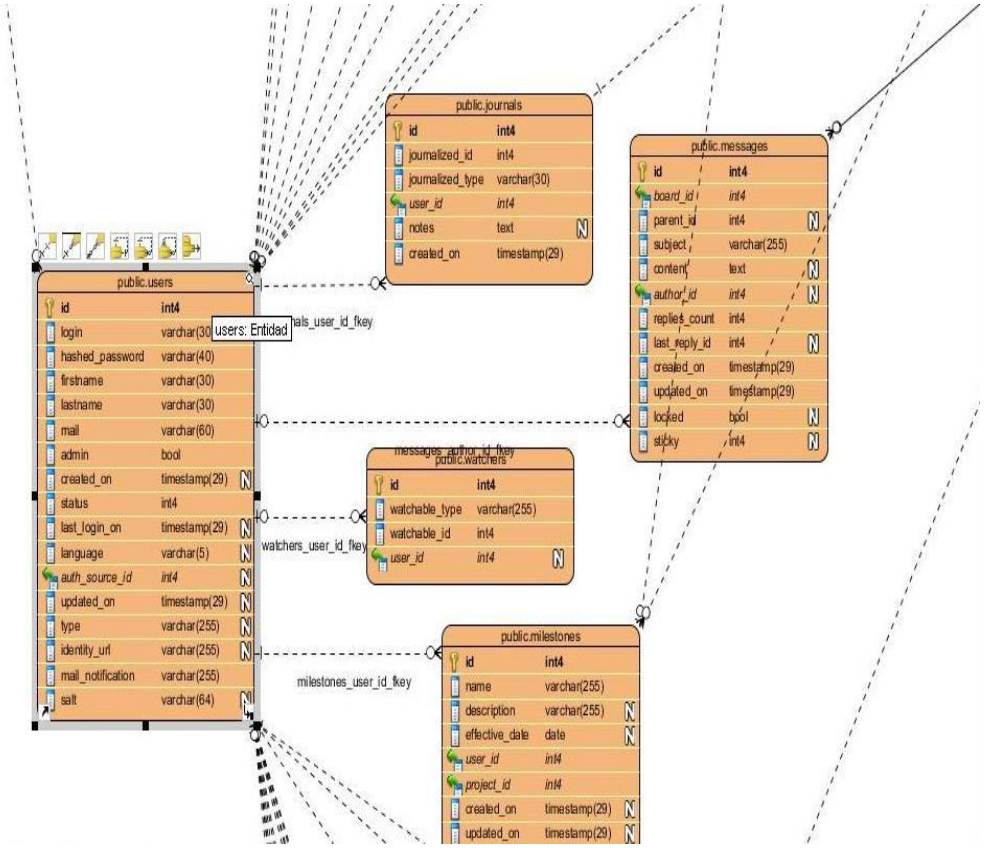

Figura 6 Relaciones de la tabla user (usuario)

En la base de datos se almacenan peticiones, estas peticiones son pertenecientes a un proyecto determinado. Por tal razón también constituye de gran importancia contemplar la tabla issues (peticiones) dentro de la base de datos del GESPRO **(Figura 7)**.

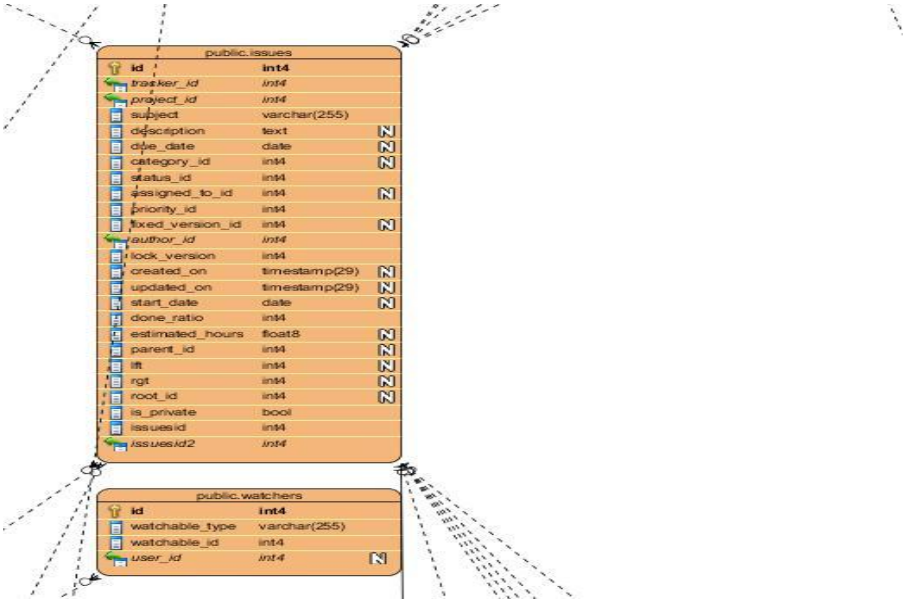

Figura 7 Relaciones de la tabla peticiones

Las tablas proyectos, usuario y peticiones en la base de datos del GESPRO son la base a partir de las cuales surgen las otras tablas. Estas tres tablas se relacionan entre sí, formando una relación mediante el patrón Entidad-Atributo-Valor (EAV). Estas tablas serían las entidades, surgiendo así dos tablas importantes en la base de datos, las custom fields y las custom values. Estas dos tablas guardan los atributos y los valores de las tablas entidades **(Figura 8)**.

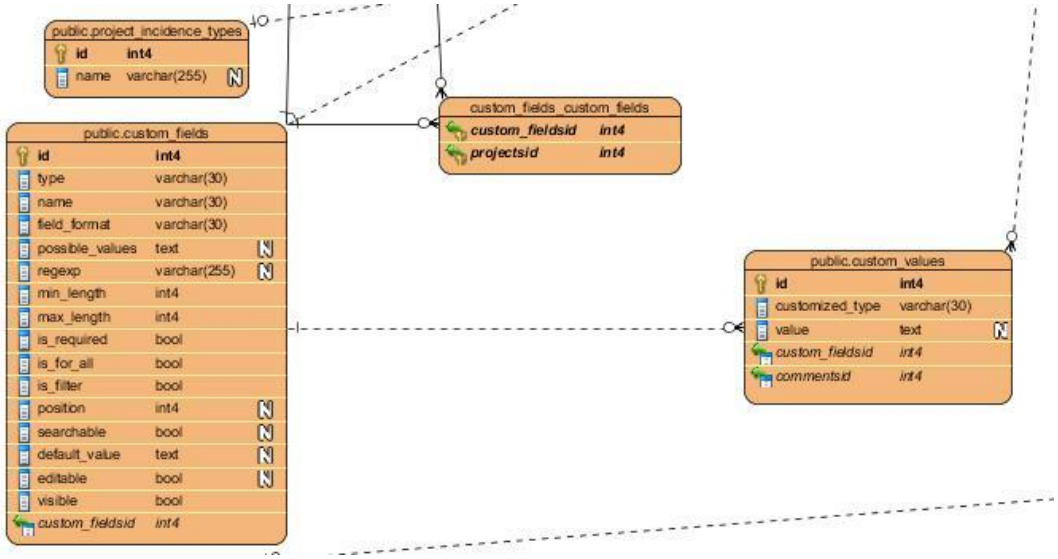

Figura 8 Relación de las tablas custom fields y custom values

En los proyectos también se toman acuerdos, se crean wiki[47], se lleva una línea base, se tiene un control de los campos personalizados, así como la autenticación y creación de usuarios. Todas estas acciones se registran en la base de datos de GESPRO y cada una de ellas tiene una tabla en las cuales se guarda toda la información.

Las tablas tienen campos en común y estos campos son las funcionalidades que se realizan en cada paso que se da en el GESPRO. Entre los campos en común que presentan las tablas se encuentran los identificadores, que son de gran importancia para diferenciar una tabla de otra, así como saber qué vinculación tiene una tabla con otra tabla. Los identificadores hacen función de llaves primarias como se ejemplifica en la **figura 9** y foráneas en dependencia de su funcionalidad en las tablas. Como característica esencial se tiene que los identificadores tienen que ser propios de la tabla y no se pueden repetir en ninguna otra. Estos cuentan con un nombre que también los diferencian de las demás.

[47] *Es un sitio web cuyas páginas pueden ser editadas por múltiples voluntarios a través del navegador web. Los usuarios pueden crear, modificar o borrar un mismo texto que comparten.*

No	Nombre de la tabla	Llave Primaria
1	agreement_statuses	agreement_statuses_pkey (id)
2	agreements	agreements_pkey (id)
3	answer_continuous_polls	answer_continuous_polls_pkey (id)
4	answer_open_polls	answer_open_polls_pkey (id)
5	asistencias	asistencias_pkey (id)
6	attachments	attachments_pkey (id)
7	auth_sources	auth_sources_pkey (id)
8	baselines	baselines_pkey (id)
9	boards	boards_pkey (id)
10	business_processes	business_processes_pkey (id)
11	business_processes_process_services	business_processes_process_services_pkey (id)
12	changes	changes_pkey (id)
13	changesets	changesets_pkey (id)
14	comments	comments_pkey (id)
15	custom_fields	custom_fields_pkey (id)
16	custom_values	custom_values_pkey (id)
17	data_source_connections	data_source_connections_pkey (id)
18	data_source_project_trans	data_source_project_trans_pkey (id)
19	data_source_projects	data_source_projects_pkey (id)
20	data_sources	data_sources_pkey (id)
21	departments_organization	departments_organization_pkey (id)
22	developer_entity	developer_entity_pkey (id)

Figura 9 Llaves primarias de cada tabla

En el modelo de datos se identificaron y seleccionaron las llaves primarias de cada tabla de la base de datos **(Figura 9).** Estas son de gran importancia en las tablas para poder relacionar el modelo físico y lógico de la base de datos.

También se realizó una ubicación física de los archivos de la base de datos **(Figura 10),** donde los nombres están encabezados por las siglas “DDL” (data definition language, en ingles) y el nombre de la acción, ejemplo: DDL_create_user_preference, realizando una descripción de cada ubicación física.

No	Nombre	Descripción
1.	DDL_issue_move	Mover las peticiones
2.	DDL_issue_add_note	Adicionar notas a las peticiones
3.	DDL_export_pdf	Exportar en pdf las peticiones
4.	DDL_issue_start_date	
5.	DDL_calendar_and_activity	Actividad y calendario de las peticiones
6.	DDL_create_journals	Crear las actividades relacionadas a los proyectos
7.	DDL_create_user_preferences	Crear las preferencias de los usuarios
8.	DDL_add_hide_mail_pref	Adicionar preferencias
9.	DDL_create_comments	Crear los comentarios
10.	DDL_add_news_comments_count	Adicionarnuevoscomentarios
11.	DDL_add_comments_permissions	Adicionar los permisos para comentar
12.	DDL_create_queries	Crear las consultas de los proyectos
13.	DDL_add_queries_permissions	Adicionar permisos para las consultas de los proyectos
14.	DDL_create_repositories	Crear los repositorios
15.	DDL_add_repositories_permissions	Adicionar los permisos para los repositorios
16.	DDL_create_settings	Crear las configuraciones
17.	DDL_set_doc_and_files_notifications	Seleccionar los documentos de los campos notificados

Figura 10 Ubicación física de los archivos

CONCLUSIONES DEL CAPÍTULO

En la vista de arquitectura de datos del GESPRO se almacena información de tipo datos, la cual es de origen interno, el destino de las fuentes de información es de almacenamiento y procesamiento temporal, y el volumen de la base de datos es de tamaño medio.

El sistema GESPRO cuenta en su vista de arquitectura de datos con 11 escenarios, los cuales a partir de los patrones de soluciones propuestos poseen, indistintamente, mejoras en los atributos de calidad: eficiencia, fiabilidad, mantenibilidad, confiabilidad y funcionalidad.

La base de datos de GESPRO cuenta con 350 términos definidos en idioma inglés, junto con la descripción de cada uno de ellos y los valores que toman, posibilitando esto evitar conflictos a la hora de insertar los datos en el sistema.

Se realizó el modelo de datos por el cual se va a diseñar la base de datos del GESPRO, lo que permitirá que el modelado de la base de datos sea lo más eficiente posible.

Capítulo 3: Validación de la arquitectura propuesta para la vista de datos del GESPRO 12.05

La arquitectura en un software después que es definida e implantada es muy difícil de cambiar, ya que es la columna vertebral de todo producto informático y alterar una parte de ella equivale a tener que alterar casi todo el sistema. Por tal razón las pruebas de validación se hacen necesarias a la hora de implantar la arquitectura para evitar así los errores que puedan surgir con el diseño de esta si no cumple con las necesidades que se definen inicialmente. En el presente capítulo se hace referencia a la evaluación de la arquitectura, su necesidad; así como al método, la técnica y el instrumento con que se realizó la validación.

3.1 NECESIDAD DE VALIDAR UNA ARQUITECTURA

La arquitectura de software constituye la base fundamental que guía el trabajo de implementación y desarrollo de un producto. Realizar su correcto diseño implica que se minimicen los posibles riesgos en el sistema tras la culminación del producto, y que el mismo cumpla con las necesidades descritas en el problema que se desea satisfacer.

Según Kazman[48], *"la garantía de una correcta arquitectura cumple un papel fundamental en el éxito general del proceso de desarrollo y en cumplimiento de los atributos de calidad del sistema. Garantizar la calidad del producto está relacionado con avalar la calidad de la arquitectura y esto es posible si se realiza una evaluación de la misma."* **(17)(37)**

El autor Omar Salvador Gómez plantea que: *"El propósito de realizar evaluaciones a la arquitectura, es para analizar e identificar riesgos potenciales en su estructura y sus propiedades, que puedan afectar al sistema de software resultante, verificar que los requerimientos no funcionales estén presentes en la arquitectura, así como determinar en qué grado se satisfacen los atributos de calidad[49]. Cabe señalar que los requerimientos no funcionales también son llamados atributos de calidad"* **(22)**.

Cuando se evalúe la arquitectura de datos del sistema de gestión (GESPRO) no se define si esta es buena o si es mala para desarrollarlo. Lo que se hace es verificar si esta cumple con los objetivos por lo

[48] *Importante arquitecto de software.*
[49] *"Especifican los criterios para juzgar la operación de un sistema en lugar de su comportamiento específico"*

cual fue diseñada, si presenta la calidad necesaria para desarrollar el software o si es necesario realizar cambios y modificaciones para lograr un mejor rendimiento del producto.

La vista de la arquitectura de datos del sistema de gestión de proyecto de la Universidad de las Ciencias Informáticas es importante validarla antes de ponerla en práctica para:

1. Verificar si se cumplieron las tareas trazadas para darle cumplimiento al objetivo de la investigación.
2. Comprobar si con la arquitectura diseñada se eliminaron las dificultades del sistema en la versión 11.05.
3. Probar si la descripción de los escenarios, el diccionario de datos y el modelo de datos presentan calidad para implementar la base de datos.

Teniendo en cuenta la importancia de validar la arquitectura del GESPRO en su versión 12.05, es importante saber en qué etapa se podrá evaluar esta arquitectura para poder determinar con mayor exactitud las posibles deficiencias. **(18)**

3.2 ¿CUÁNDO VALIDAR UNA ARQUITECTURA?

La arquitectura puede ser evaluada en cualquier momento de desarrollo. El autor Omar Salvador Gómez, tras una investigación realizada define dos "reglas de oro" para determinar el momento en el cuál realizar la evaluación: **(22)**

> Regla 1: Cuando el equipo de desarrollo comienza a tomar decisiones que afectan directamente a la arquitectura.
>
> Regla 2: Cuando el costo de no tomar estas decisiones podría ser mayor que el costo de realizar una evaluación.

Por otro lado, algunos autores (Clements, Kasman,[50] entre otros) plantean que es más común realizarla luego de haberla especificado totalmente y el software aun no se ha implementado. **(37)**

Coincidiendo en parte con esto **(21)**, se definen dos variantes útiles para realizar las evaluaciones, una de ella es la *evaluación temprana* y la otra es la *evaluación tardía*.

[50] *Especialistas en la arquitectura de software.*

La evaluación temprana

No es necesario que la arquitectura esté completamente diseñada para realizar la evaluación, y esto incluye desde las etapas tempranas de diseño hasta su desarrollo.

La evaluación tardía

Esta se ejecuta cuando la arquitectura está establecida y la implementación se ha completado. Este es el caso general que se presenta en un sistema ya implementado y desarrollado.

En el caso de esta investigación, la evaluación de la arquitectura se realiza en la forma de evaluación temprana, ya que se inicia la evaluación en el momento en que la arquitectura se encuentra completamente definida mas el sistema, aun no se ha implementado.

En la evaluación de la arquitectura, los atributos de calidad juegan un papel fundamental. A continuación se dará una definición precisa sobre los atributos de calidad.

3.3 ¿QUÉ SON LOS ATRIBUTOS DE CALIDAD?

Los atributos de calidad son características o atributos que se afectan en un sistema, ya sean positiva o negativamente. Se puede decir que son servicios que brinda el sistema a los usuarios y por tal razón se desean perfeccionar para brindar una mejor calidad. En la **figura 11** se muestra una tabla con algunos atributos de calidad y la descripción de cada uno de ellos.

Atributo de Calidad	Descripción
Disponibilidad	Es la medida de disponibilidad del sistema para el uso
Confidencialidad	Es la ausencia de acceso no autorizado a la información
Funcionalidad	Habilidad del sistema para realizar el trabajo para el cual fue concebido
Desempeño	Es el grado en el cual un sistema o componente cumple con sus funciones designadas, dentro de ciertas restricciones dadas, como velocidad, exactitud o uso de memoria.
Confiabilidad	Es la medida de la habilidad de un sistema a mantenerse operativo a lo largo del tiempo
Seguridad externa	Ausencia de consecuencias catastróficas en el ambiente. Es la medida de ausencia de errores que generan pérdidas de información
Seguridad interna	Es la medida de la habilidad del sistema para resistir a intentos de uso no autorizados y negación del servicio, mientras se sirve a usuarios legítimos

Figura 11. Atributos de Calidad (22)

Para mejorar los atributos de calidad se hace necesario el diseño de una nueva versión del sistema y por tal razón se realiza una nueva arquitectura de software la cual contemple las mejoras que se le desean realizar al software en cuestión.

La calidad de software se define como el grado en el cual el software posee una combinación deseada de atributos. Tales atributos son requerimientos adicionales del sistema, que hacen referencia a características que éste debe satisfacer, diferentes a los requerimientos funcionales. **(19)**

Los atributos de calidad pueden clasificarse en dos categorías: **(17)**

- Observables en tiempo de ejecución: Son aquellos atributos que cuando el software está en proceso de ejecución, son determinados fácilmente por el usuario.
- No observables en tiempo de ejecución: Son los atributos que se establecen mediante el desarrollo del sistema. No hay que esperar a que el software se ejecute para identificarlos.

En la solución a la vista de la arquitectura de datos de GESPRO para la versión 12.05, se describen patrones de solución para los escenarios que intervienen en la vista. Estos patrones, la definición del diccionario de datos y el modelado de la base de datos, permiten el mejoramiento de atributos de calidad tales como:

- Eficiencia
- Funcionalidad
- Fiabilidad
- Mantenibilidad
- Confiabilidad

Estos atributos hacen que el rendimiento de la base de datos del sistema de gestión de proyecto de la Universidad de las Ciencias Informáticas tenga la calidad necesaria para brindar un mejor servicio.

3.4 VALIDACIÓN DE LA ARQUITECTURA

En el sistema de gestión de proyectos, GESPRO, implementado en la Universidad de la Ciencias Informáticas para la gestión de los proyectos que se desarrollan en dicho centro, se encontraron dificultades que conllevaron a la elaboración y construcción de una nueva versión, la 12.05. Para la misma se diseñó una nueva arquitectura con el propósito de eliminar las deficiencias antes encontradas.

Para detectar las deficiencias en la versión 11.05 de aplicó una encuesta **(Anexo 1)** a especialistas de GESPRO, donde el 100% de los encuestados coincidieron en que el sistema presentaba problemas con:

1- La definición de los escenarios y de sus patrones de solución.

2- El diccionario de datos y las definiciones de los conceptos de las variables o términos de la base de datos.

3- El modelo de datos del sistema.

Para solucionar estas deficiencias, se definió la vista de la arquitectura de datos del sistema de gestión de proyectos de la Universidad de las Ciencias Informáticas y para poder implantar esta propuesta se hace

necesaria la validación de la misma. La validación se hizo basada en caso de estudio, aplicando la técnica Delphi [51]y como instrumento se utilizó la encuesta.

Para poder aplicar la arquitectura del sistema fue necesario comprobar si se satisficieron las necesidades y se corrigieron las deficiencias encontradas en la versión 11.05. En la validación de esta propuesta de solución se aplicó el método delphi o método de experto. A continuación se darán algunos argumentos para explicar en qué consiste este tipo de validación y cómo se realizó la misma.

3.4.1 Técnica de validación e instrumento a utilizar

Para realizar la evaluación o validación de la vista de la arquitectura de datos del sistema de gestión de proyecto (GESPRO) de la Universidad de las Ciencias Informáticas (UCI) se aplicó una técnica muy conocida e implementada en el mundo. La técnica aplicada es la cualitativa mediante encuesta o cuestionarios y la técnica Delphi; es una de las muchas técnicas que existe para la validación de la arquitectura de un sistema informático. **(Figura 12)**. **(23)**

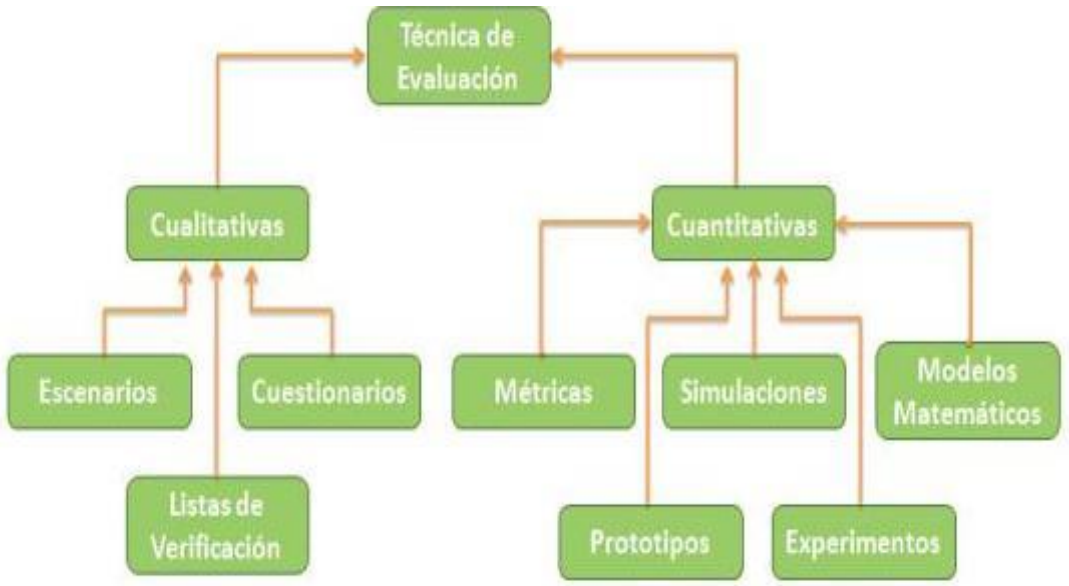

Figura 12. Técnicas de evaluación

3.4.2 El método Delphi

[51] *Es una metodología de investigación multidisciplinar para la realización de pronósticos y predicciones.*

La **t**écnica Delphi entra en la familia de los métodos de pronóstico y habitualmente se clasifica dentro de los métodos cualitativos o subjetivos.

Eneko Astigarraga[52] (2003-2004) de la Universidad de Deusto señala que*:"Delphi consiste en la selección de un grupo de expertos a los que se les pregunta su opinión sobre cuestiones referidas a acontecimientos del futuro. Las estimaciones de los expertos se realizan en sucesivas rondas anónimas, con el objetivo de tratar de conseguir consenso, pero con la máxima autonomía por parte de los participantes. Por lo tanto, la capacidad de predicción del Delphi se basa en la utilización sistemática de un juicio intuitivo emitido por un grupo de expertos. Es decir, el método Delphi procede por medio de la interrogación a expertos con la ayuda de cuestionarios sucesivos, a fin de poner de manifiesto convergencias de opiniones y deducir eventuales consensos. La encuesta se lleva a cabo de una manera anónima (…) para evitar los efectos de líderes".* **(36)**

Faces del método delphi

Este método Delphi presenta 4 faces fundamentales que hacen que la aplicación del método sea más fácil y entendible para las personas que decidan aplicarlo en sus soluciones.

1. Formulación del problema.
2. Elección de expertos.
3. Elaboración y lanzamiento de los cuestionarios (en paralelo con la fase 2).
4. Desarrollo práctico y explotación de resultados.

3.4.3 ¿Cómo se validó?

Lo primero que se realizó fue elaborar las preguntas que se van a comprobar con los expertos seleccionados.

Se confeccionaron 5 preguntas que permitieran comprobar la arquitectura propuesta, teniendo en cuenta los atributos de calidad que intervienen en la solución **(Anexo 2)**.

[52] *Profesor de Prospectiva en la Universidad Deusto. Facultad de Empresariales de San Sebastián*

Se seleccionaron 7 expertos teniendo en cuenta los criterios siguientes:

- Ser graduado de Nivel Superior.
- Tener un año de experiencia como mínimo en el sistema de gestión de proyectos (GESPRO).
- Presentar conocimientos acerca de la base de datos de GESPRO.

Seleccionado los expertos y el cuestionario confeccionado, se dio paso a que los expertos realizaran la encuesta. Se les explicó que le dieran una valoración del 1 al 5 en dependencia de la influencia de los atributos de calidad que intervienen en la vista de la arquitectura de datos y la solución propuesta para su mejoramiento, que es la base de las preguntas confeccionadas.

Una vez aplicada la encuesta se analizaron los resultados expuestos (**Figura 13**) por los expertos.

2	Criterios	E1	E2	E3	E4	E5	E6	E7	Rj	$(R-Rmedia)^2$
3	Crit 1	5	5	3	3	4	4	4	28	0
4	Crit 2	4	4	3	4	3	5	5	28	0
5	Crit 3	5	4	4	3	5	4	4	29	1
6	Crit 4	4	4	5	3	4	2	4	26	2,25
7	Crit 5	5	4	5	4	3	4	4	29	0
8									S =	3,25

Figura 13. Resultados de la encuesta

A continuación se aplica la formula de Kendal y la de Chi cuadrado (X^2) [53]para probar el grado de concordancia en las valoraciones realizadas por los expertos.

Kendal plantea que el Coeficiente de Concordancia es igual a:

$W= 12S / E^2 * (N^3-N)$

Donde S: Es la suma de los cuadrados de las desviaciones observadas de la media de Rj (rangos), esto es:

[53] *Fórmulas que se aplican para ver el nivel de concordancia de algo en cuestión*

$S= \Sigma^{n}_{j-1}(Rj - M)^2$ donde $M= \Sigma^{n}_{j-1}Rj/N$

N: Número de preguntas realizadas a los expertos.

E: Cantidad de expertos.

M: Suma de los rangos dividido entre la cantidad de preguntas realizadas

W: Coeficiente de certeza.

Aplicando las fórmulas se obtiene que:

S= 3.25 y W= 0.003

W expresa el grado de concordancia entre los siete expertos al dar un orden evaluativo a las preguntas sometidas a valoración. Este coeficiente siempre será positivo y su valor estará comprendido en el rango de 0 a 1

Para calcular el Chi Cuadrado Real se aplica la siguiente fórmula:

$X^2 = E * (N - 1) * W$ por lo que sustituyendo y calculando se tiene que:

$X^2 = 0.16260163$. Si X^2 real < X^2 (∞, N-1) entonces existe concordancia entre los expertos.

El Chi cuadrado calculado se compara con los valores de tablas estadísticas dadas, con ∞=0.05 para un nivel de confianza del 95%.

X^2 real < X^2 (∞, N-1)

0.16260163 < X^2 (0.05, 5 - 1)

0.16260163 < 9,48772904

Por lo tanto, se puede concluir que existe concordancia entre los expertos. Por lo que la vista de la arquitectura de datos del sistema de gestión de proyecto (GESPRO) de la Universidad de las Ciencias Informáticas está en condiciones de ser implantada en el sistema.

CONCLUSIÓNES DEL CAPÍTULO

En el presente capitulo se realizó una validación de la vista de la arquitectura de datos del sistema de gestión de proyecto GESPRO, donde se llegó a la conclusión que la vista de la arquitectura de datos del GESPRO en su versión 12.05, ha resuelto los problemas encontrado en la versión 11.05 del sistema y está en condiciones de ser implantada, ya que presenta la calidad necesaria para ponerla en práctica según la concordancia de los especialistas.

CONCLUSIONES GENERALES

Una vez realizado un estudio acerca de los diferentes modelos de la arquitectura y de las vistas que componen la misma, se decidió escoger el modelo 9+1 vista de la arquitectura de software propuesto por la Universidad de las Ciencias Informáticas (UCI) como modelo a seguir en el diseño de la vista de la arquitectura de datos del sistema de gestión de proyectos (GESPRO) en este centro de altos estudios. De igual forma se escogió la herramienta Visual Paradigm para el modelado de la base de datos y se propone que se utilice el PostgreSQL como herramienta gestora de la base de datos del GESPRO.

La definición correcta de los 11 escenarios identificados en el sistema GESPRO con sus respectivos patrones de solución proyectados en función de mejorar los atributos de calidad, más la confección del diccionario de datos y del modelo de datos, conllevaron a que se solucionaran los problemas identificados cumpliéndose así el objetivo de la investigación.

Con la investigación se eliminaron las insuficiencias detectadas en la versión 11.05 quedando la vista de la arquitectura de datos con la calidad requerida para implantarla; siendo esto validado a través del método Delphi.

RECOMENDACIONES

Añadir otras funcionalidades a la base de datos del GESPRO para garantizar una mejor confiabilidad.

Utilizar la presente investigación como material de consulta para realizar otras versiones del sistema de gestión o investigaciones similares.

REFERENCIAS BIBILOGRÁFICAS

1. Perry, Dewayne y Wolf, Alexander. *Foundations for the study of software architecture.* s.l.: ACM SIGSOFT Software Engineering, 1992.

2. *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft.* Billy, Carlos y Kicillof, Nicolás**.**2004.

3. *"Introducción a la Arquitectura de Software".* Reynoso, Carlos Billy**.** 2005.

4. *EL ROL DEL ARQUITECTO DE SOFTWARE.* Julio César Rosales Colindres. *Facultad de Ingeniería – Revista Ingeniería Primero No. 19 – Octubre, 2010 - Pags.18 -22.*

5. *Rational Unified Process. Rational Software, IBM, 2003.http://www-306.ibm.com/software/awdtools/rup/*

6. *EL PROCESO DE INVESTIGACIÓN CIENTÍFICA Rolando Alfredo Hernández León y Sayda Coello González*

7. IEEE Std 1471-2000, 2000.

8. BASS, "*Arquitectura de Software en la Práctica (Software Architecture in Practice)"* 2nd edición, 2003

9. Roger Pressman, *"Ingenieroa de Software: Un aporte práctico"*, 2005

10. REYNOSO, 2004

11. Bass, Clements y Kazman, 2003

12. Christopher Alexander, "*The Oregon Experiment"*, 1977.

13. Estilos y Patrones en la Estrategia de Arquitectura de Microsoft

14. Introducción a la Arquitectura de Software. Carlos Billy Reynoso. Marzo 2004.

15. Kenneth Kendal, *"Análisis y diseño de un sistema",* México, 1993

16. Erika Camacho, Fabio Cardeso, Gabriel Nuñez. Arquitecturas de Software. 2004.

17. Bosch, Jan. Design and Use of Software Architectures. s.l. : Addison Wesley, 2000.

18. Clements, Paul, Kasman, Rick y klein, Mark. Evaluating Software Architecture: Methods and Case Studies. s.l. : Addison-Wesley, 2002.

19. Barbacci, Mario, y otros. Quality Attributes. Technical Report. Carnegie Mellon University. Pittsburgh, Pennsylvania 15213 : s.n., 1995.

20. Puebla, Yoan Arlet Carrascoso, Gómez, Enrique Chaviano y Vega, Anisleydi Céspedes. Procedimiento para la evaluación de arquitecturas de software basadas en componentes. [En línea] 2009.

21. Salvador Gómez, Omar. Evaluando Arquitecturas de Software. Parte 1. Panorama General. s.l. : Brainworx S.A, 2007

22. RICARDO ARTURO OSORIO ROJAS. Profesor de la Universidad de Oriente. Cuba.

23. Billy Reynoso, Carlos. Introducción a la Arquitectura de Software . Buenos Aires : Microsoft Press, 2004.

24. Perry, D., & Wolf, A. Foundations for the Study of Software Architecture. ACM Sigsoft - Software Engineering Notes. Obtenido el 30-05-2002. Vol. Vol 17. No. 4, www.ics.uci.edu/~taylor/ICS221/papers/swa-sen.pdf.

25. Abdurazik., Aynur. Suitability of the UML as an Architecture Description Language with applications to testing. s.l. : George Mason University, Febrero de 2000. Reporte ISE-TR-00-01.

26. Kruchten., Philippe. The 4+1 View Model of Architecture. s.l. : IEEE Software, Noviembre de 1995. pp. pp. 42-50.

27. Bass, Len, Clements, Paul and Kazman, Rick. Software Architecture in Practice . s.l: Addison Wesley, 2003. 0-321-15495-9.

28. Clements, Paul. A Survey of Architecture Description Languages. Proceedings of the International Workshop on. Alemania : s.n., 1996.

29. Grady Booch, James RumbaugheIvar Jacobson. El Lenguaje Unificado de Modelado. Madrid : Addison-Wesley, 1999.

30. Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides. Design Patterns: Elements of reusable objectoriented. s.l. : Addison-Wesley, 1995.

31. Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad y Michael Stal. Pattern-oriented software. A system of patterns. s.l. : John Wiley & Sons, 1996.

32. Perry, D., & Wolf, A. Foundations for the Study of Software Architecture. ACM Sigsoft - Software Engineering Notes. Obtenido el 30-05-2002. Vol. Vol 17. No. 4, www.ics.uci.edu/~taylor/ICS221/papers/swa-sen.pdf.

33. MsC René Lazo Ochoa, *"Modelo de referencia para el desarrollo arquitectónico de sistemas de software en dominios de gestión"*, Universidad de la Ciencias Informáticas, 2011.

34. Laboratorio de soluciones e investigaciones avanzadas en la gestión de proyecto. Universidad de las Ciencias Informáticas, Vicerrectoría de formación: *"Normas para la redacción, presentación y defensa de tesis"*. La Habana, 2012.

35. Eneko Astigarraga: *"El Método Delphi"*, Universidad de Deusto, 2003-2004.

36. Kazman, R, Clements, Paul y Klain, M. 2005. *Evaluating Software Architectures.Methods and case studies. 1ra Edicición.* s.l. : Adison-Wesley Professional, 2005. pág. 368. ISSN: 978-0201704822 .

37. Erika Camacho, Fabio Cardeso, Gabriel Nuñez. Arquitecturas de Software.2004.

38. Maestre Martínez, Roberto. "*Recursos electrónicos para historiadores. Bases de datos, representación y análisis de redes complejas*". 2011.

39. Valdiviezo Serrano, Patricia Alexandra Guacho Minta, María Alejandra. *"Análisis Comparativo de Tecnologías de Aplicaciones Web en el Entorno JSF y ADF. Caso Práctico: IESS de Riobamba - Chimborazo."* 2012.

40. Ernest Abadal Falgueras. *"Gestión de proyectos en información y documentación".* España, 2004.

BIBLIOGRAFÍAS

1. **Bosch, J. 2000.** *Design & Use of Software Architectures*. s.l. : Adison-Wesley, 2000.
2. **Carriere, J, Kazman, R y Woods, S. 2007**. *Toward a Discipline of Scenario based Architectural Engineering*. s.l. : Software Engineering Institute, Carnegie Mellon University, 2007. págs. 5-33. ISSN: 1573-7489.
3. **Gómez, Omar Salvador. 2007.** E*valuando Arquitecturas de Software. Parte 1*. Panorama General. México : México: Brainworx S.A, 2007.
4. **Kazman, R, Bass, L y Clements, P. 2003**. *Software Architecture in Practice. 2da Edición*. s.l. : Addison-Wesley Professional, 2003. pág. 560. ISSN: 0321154959 .
5. Camacho E., Cardeso F. y Núñez G. (2004). Guía de estudio de la Arquitectura de Software.
6. Hernández G. T. y Piquera V. L A. (1997). Manual de usuarios del MicroSet NT. Camagüey Cuba.
7. Jacobo I., Booch G. y Rumbaugh J. (2004). El Proceso Unificado de Desarrollo de Software. Volumen 1. La Habana Cuba: Félix Varela.
8. King G. C. (2005). Iberneig in Action. Estados Unidos de América: Manning blications.
9. Larman C. (2004). UML y Patrones. Introducción al análisis de diseño orientado a objetos. Parte 2. La Habana Cuba: Editorial Félix Varela.
10. Pressman R. (2005). Ingeniería del Software. Un enfoque práctico. Parte 2. La Habana Cuba: Félix Varela.
11. Reynoso C. y Kicillof N. (2004). Electronic Source: Estilos y Patrones en la Estrategia de Arquitectura de Microsoft. Universidad de Buenos Aires. <http://www.microsoft.com/spanish/msdn/arquitectura/roadmap_arq/style.asp#1 > [Consulta: 2 de diciembre del 2006].

ANEXOS

Anexo 1 *(Encuesta realizada a los especialistas de GESPRO para detectar deficiencias en la versión 11.05)*

	Especialista 1	**Especialista 2**	**Especialista 3**	**Especialista 4**	**Especialista 5**
1- **Existen problemas a la hora de almacenar los datos**	No Existen	No Existen	No Existen	No Existen	No Existen
2- El control y tratamiento de los datos que se maneja se hace engorroso	No	No	No	No	No
3- Se dificulta las conexiones de las bases de datos (BD) de la universidad con la herramienta	No	No	No	No	No
4- El diccionario de datos está definido	No	No	No	No	No
5- Existe conflictos a la hora de manipular los datos en la herramienta	No	No	No	No	No
6- La descripción de los escenarios y patrones de solución está definida	No	No	No	No	No
7- Las conceptualizaciones que se identifican en la base de datos son de buena calidad	Si	Si	Si	Si	Si
8- El modelo lógico y físico de la base de dato está	No	No	No	No	No

implementado					

Anexo 2 *(Serie de preguntas realizada a los especialistas de GESPRO para aplicar el método delphi)*

Pregunta 1

Eficiencia: Este atributo de calidad se mejora considerablemente con la aplicación de los escenarios de la vista de datos: Manejo de pool de conexiones, Tratamiento con estructuras de árboles desde la base de datos, Procesamiento de los datos, Configuración básica del servidor de datos e Indexado de la base de datos. También con el desarrollo del modelo de datos se pretende una mejor eficiencia del sistema. Estos escenarios y el modelo de datos permiten que:

- Todas las conexiones sean controladas de manera centralizada, de forma tal que el sistema pueda auditar y realizar acciones que mejoren el rendimiento y la transmisión de datos.
- La base de datos conozca a todos los hijos pertenecientes a un determinado padre y viceversa.
- Se apliquen técnicas de procesamiento de los datos para el tratamiento de estos en la base de datos, logrando un mejor rendimiento de la misma.
- Se conozcan bien las relaciones entre las tablas de la base de datos evitando presentar problemas a la hora de elaborarla.

1) ___ 2) ___ 3) ___ 4) ___ 5) ___

Pregunta 2

Mantenibilidad: Este atributo de calidad se mejora considerablemente con la aplicación de los escenarios de la vista de datos: Transparencia entre la capa de negocio y la capa de datos, Configuración básica del servidor de datos, Salvas y respaldo de la base de datos y Política de evolución y crecimiento de los datos. Estos escenarios permiten que:

- El sistema cuente con un mecanismo de mapeo relacional de objeto que posibilita persistencia compuesta, carga perezosa, actualización y modificación sincronizada; tanto en el modelo mapeado como en el modelo relacional.
- Exista un equipo encargado de la administración y tuning de las bases de datos.
- Se realice una salva periódicamente a la base de datos, para que en caso de algún fallo el sistema no se pierda la información.

1) ___ 2) ___ 3) ___ 4) ___ 5) ___

Pregunta 3

Funcionalidad: Este atributo de calidad se mejora considerablemente con la aplicación de los escenarios de la vista de datos: Administración de transacciones, Réplica de datos, Procesamiento de los datos,

Configuración básica del servidor de datos y con el escenario Salvas y respaldo de la base de datos. También con el desarrollo del diccionario de datos y del modelo de datos se pretende una mejor funcionalidad del sistema. A continuación se muestran algunas acciones que mejoran la funcionalidad del sistema.

- Se realiza una salva periódicamente a la base de datos, para que en caso de algún fallo el sistema no pierda la información.
- Se permite que las operaciones de modificación sobre el modelo de dominio sean transaccionales por defecto.
- Se identifican bien las relaciones entre las tablas de la base de datos evitando así presentar problemas a la hora de elaborarla y que su funcionalidad sea la más eficiente.
- Se obtiene una descripción detallada de los términos de la base de datos, para de esta manera garantizar una buena funcionalidad de la base de datos.

1) ___ 2) ___ 3) ___ 4) ___ 5) ___

Pregunta 4

Fiabilidad: Este atributo de calidad se mejora considerablemente con la aplicación de los escenarios de la vista de datos: Administración de transacciones, Configuración básica del servidor de datos, Salvas y respaldo de la base de datos y Política de evolución y crecimiento de los datos. También con el desarrollo del diccionario de datos se pretende la fiabilidad del sistema. A continuación se muestran algunos puntos por lo cual el sistema de gestión es más fiable.

- Se permite que las operaciones de modificación sobre el modelo de dominio sean transaccionales por defecto y que se configuren las notificaciones ante la posibilidad de fallas en la transacción por un mecanismo central de notificaciones.
- Existe un equipo encargado de la administración y tuning de las bases de datos.
- Se obtiene una descripción detallada de los términos de la base de datos, para de esta manera garantizar que la fiabilidad a la hora de insertar los datos sea óptima.

1) ___ 2) ___ 3) ___ 4) ___ 5) ___

Pregunta 5

Confiabilidad: Este atributo de calidad se mejora considerablemente con la aplicación del escenario de la vista de datos: Concurrencia en el acceso a dato. También con el desarrollo del diccionario de datos y del modelo de datos se pretende la confiabilidad en el sistema. A continuación se muestran algunos puntos por lo cual el sistema de gestión es más confiable.

- Se permite configurar el esquema de concurrencia que se desee sobre las entidades (conceptos) o estructuras de entidades del esquema de persistencia de una solución específica.

- A cada tabla (no a los nomencladores) se le agrega un campo versión que permite el control de las modificaciones.
- Se obtiene una descripción detallada de los términos de la base de datos, para de esta manera garantizar una mayor confiabilidad a la hora de insertar los datos.
- Se conocen bien las relaciones entre las tablas de la base de datos evitando así presentar problemas a la hora de elaborarla y que con esto se haga más confiable la construcción de la base de datos.

1) ___ 2) ___ 3) ___ 4) ___ 5) ___