UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
DIRECCIÓN DE FORMACIÓN POSTGRADUADA

Modelo para el diseño y evaluación de la arquitectura de los sistemas de gestión de información biomédica

MEMORIA PRESENTADA

EN OPCIÓN AL TÍTULO DE

MÁSTER EN INFORMÁTICA APLICADA

Autor: *Ing.* Yusdenis Sánchez Perodín

Tutores: *M.Sc.* Maikel Yelandi Leyva Vázquez
M.Sc. Yanet Villanueva Armenteros

Ciudad de La Habana, Julio de 2010

DECLARACIÓN JURADA DE AUTORÍA

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste se firma la presente a los _____ días del mes de _____ del año _____.

_____
Ing. Yusdenis Sánchez Perodín
Firma del autor

_____ _____
M.Sc. Maikel Yelandi Leyva Vázquez *M.Sc.* Yanet Villanueva Armenteros
Firma del tutor Firma del tutor

Agradecimientos

A mi familia, por apoyarme siempre y confiar en mí.
A mi novia, por los lindos momentos vividos y los que quedan por vivir.
A mis tutores, por su constante preocupación, esfuerzo y dedicación.
A mis amigos y colegas de trabajo.

...gracias.

Resumen

La definición de la arquitectura es uno de los aspectos claves para alcanzar el éxito en el desarrollo de los sistemas informáticos, esta importante tarea se puede realizar a través de la intuición, de métodos o a través de la reutilización de arquitecturas ya definidas. En la facultad seis de la Universidad de las Ciencias Informáticas se ejecutan un grupo de proyectos de desarrollo de software de gran importancia para el país, en los que se define su arquitectura a través de la intuición, lo que ha incidido en la calidad de las soluciones desarrolladas. En esta investigación se analizan un conjunto de métodos que pudieran ser utilizados para definir las arquitecturas de los sistemas, se propone un modelo como herramienta para contribuir a la solución del problema científico y se evalúa este modelo mediante su aplicación práctica en dos proyectos de la facultad.

Palabras claves: arquitectura, modelo, sistemas informáticos.

Abstract

The definition of architecture is one of the keys to success in the development of computer systems, this important task can be done through intuition, through methods that allows to define it or through the reuse of architectures already defined. In the Six School of the University of Computer Sciences are developing a group of software development projects of major importance for the country, where the architecture is defined through intuition, which has affected the quality of the developed solutions. This research analyzes a set of methods that could be used to define the architecture of systems, we propose a model as a tool to help solve the scientific problem and evaluated this model by its practical application in two software development projects in the School.

Keywords: architecture, model, computer systems.

Índice general

DECLARACIÓN JURADA DE AUTORíA **i**

Agradecimientos **iii**

Resumen **v**

Introducción **xi**

1 Fundamentación Teórica **1**

1.1 Arquitectura de Software 2

1.2 Atributos de calidad 3

1.3 Patrones o estilos arquitectónicos 4

1.4 Lenguajes de descripción de la arquitectura 7

1.5 Métodos basados en la Arquitectura de Software 9

1.6 Sistemas de gestión de información biomédica 12

1.7 Conclusiones parciales 16

2 Modelo propuesto **17**

2.1 Descripción del modelo 18

2.1.1 Determinar si los requisitos actuales son suficientes para definir la arquitectura 19

2.1.2 Definir componentes externos que serán utilizados 22

2.1.3 Seleccionar un elemento a descomponer del sistema 23

2.1.4 Identificar los conductores arquitectónicos 24

2.1.5 Seleccionar un concepto de diseño que satisfaga los conductores arquitectónicos 25

2.1.6 Asignar responsabilidades y definir interfaces 26

2.1.7 Describir la arquitectura del sistema 27

2.1.8 Verificar el cumplimiento de los requisitos 28

2.2 Técnicas y herramientas de soporte al modelo 29

2.2.1 Selección del lenguaje de descripción de la arquitectura 29

2.2.2 Técnicas de priorización de requisitos 33

2.2.3 Método visual para la selección de conductores arquitectónicos . . . 34

2.3 Conclusiones parciales 37

3 Evaluación del modelo **39**

3.1 Valoración general 40

3.2 Comparación con otros métodos 41

3.3 Aplicando el modelo 47

3.3.1 Sistema informático de Genética Médica 47

3.3.2 Sistema para la representación gráfica de árboles genealógicos . . . 52

3.4 Conclusiones parciales 54

Conclusiones **55**

Recomendasiones **57**

Referencias Bibliográficas **59**

Introducción

La arquitectura de software (ASW) es un área dentro la ingeniería de software que tal y como afirma Carlos Billy [Rey04] "... a pesar de que se acostumbra remontar sus antecedentes al menos hasta la década de 1960, su historia no ha sido tan continua como la del campo más amplio en el que se inscribe, la ingeniería de software",no es hasta la década del noventa que comienza a verse como un elemento clave en para el éxito del desarrollo de sistemas.

El proceso de desarrollo de software involucra varias fases a través de las cuales se logra obtener el sistema informático, fases que abarcan desde la descripción y captura de requisitos hasta la transferencia de tecnologías o despliegue. Dos de de las etapas claves para el éxito o fracaso de una solución son la gestión de proyectos y la definición de la arquitectura del sistema, las que presentan una alta incidencia en el costo y tiempo de los proyectos.

A través de la definición de la arquitectura se crean las bases sobres las cuales se desarrollarán los sistemas, definiéndose aquellas estructuras que formarán parte de la solución y los mecanismos de comunicación entre estas; con lo que se logra en un trabajo sincronizado y armónico que posibilita cumplir con las funcionalidades y los atributos de calidad propuestos para el sistema.

Existe un grupo de métodos que en la actualidad posibilitan definir la arquitectura de los sistemas; enfocados tanto en su diseño, descripción o evaluación. Estos constituyen la vía más indicada cuando no se cuenta con experiencias satisfactorias en trabajos previos [Kra95], proporcionan un conjunto de pasos lógicos que permiten lograr tal objetivo.

La universidad de las Ciencias Informáticas (UCI), es centro de altos estudios en los que no sólo se forman profesionales; también se encuentra inmerso en un proceso productivo constante dirigido a informatizar la sociedad cubana e incrementar los ingresos al país por conceptos de desarrollo de software. Dicha universidad está compuesta por trece facultades, dentro de estas podemos encontrar la facultad seis la cual se especializa

en tecnologías de bases de datos y el desarrollo de software de gestión de información biomédica.

Ejemplos de los proyectos que se llevan a cavo en la facultad seis son: sistema informático de genética médica, sistema de ayuda médica para la atención a la Dislipoproteinemias y herramienta para la automatización de los estudios clínico genéticos que hoy se han llevado a cabo en cuatro países del continente americano.

Los equipos de trabajo de los proyectos están dados por jóvenes recién graduados y estudiantes de pregrado de la universidad. A pesar de que con desarrollo de la actividad productiva se ha ido adquiriendo una madurez en el proceso de desarrollo de software todavía quedan muchas oportunidades de mejora.

En cuanto a la calidad de las aplicaciones desarrolladas un factor que ha influido negativamente, es el diseño de arquitecturas a través de experiencias en trabajos previos o basados en las tecnologías utilizadas; comenzando un desarrollo adelantado de las soluciones y omitiendo un estudio adecuado de los atributos de calidad, aspectos que impiden atender y resolver en los inicios del proyecto los riesgos asociados a los requisitos más críticos y a las decisiones de diseño más difíciles, que pueden comprometer el éxito del proyecto. Elementos que demuestran la existencia de una baja competencia arquitectónica [BCKK08], lo cual significa no mantener resultados arquitectónicos de manera constante y que los resultados se logren independientemente de los conocimientos de un arquitecto en particular.

Definir la arquitectura utilizando un procedimiento sistémico y evaluar la calidad de la misma asociada al cumplimiento de los requisitos, son elementos claves que elevan la probabilidad de éxito de los sistemas informáticos en etapas tempranas de su desarrollo. Lamentablemente son aspectos que no juegan un papel primordial en la facultad seis, lo cual ha provocado el desarrollo de sistemas que no cuentan con el rendimiento, modularidad, modificabilidad o escalabilidad deseado; siendo estos, atributos de calidad de elevada necesidad de cumplimiento.

Por todo lo anteriormente expuesto se plantea como *Problema Científico* ¿Cómo contribuir a un mejor cumplimiento de los atributos de calidad en los sistemas de gestión de información biomédica desarrollados en la facultad seis desde el proceso de definición de sus arquitecturas?

El *objeto de estudio* de la presente investigación es la arquitectura de software, enmarcado en el *campo de acción* el proceso de diseño y evaluación de la arquitectura de software.

Para dar solución al *Problema Científico* se plantea como *Objetivo General* "Definir un modelo para el diseño y evaluación de la arquitectura de los sistemas de gestión de información biomédica desarrollados en la facultad seis", trazándose como *Objetivos Específicos*:

- Elaborar marco teórico de la investigación.
- Desarrollar el modelo para el diseño y evaluación de la arquitectura de los sistemas de gestión de información biomédica.
- Evaluar el modelo obtenido mediante la aplicación de este en proyectos reales.

Fue identificada como *hipótesis* para la presente investigasión que realizar la definición de la arquitectura en los sistemas de gestión de información biomédica desarrollados en la facultad seis a través de un modelo que integre técnicas y métodos para este fin, contribuye al cumplimiento de los atributos de calidad en dichos sistemas.

Con vista a alcanzar los objetivos trazados se establecen siguientes *tareas*:

- Estudio de los métodos basados en la arquitectura de software y su relación con el cumplimiento de los atributos de calidad.
- Análisis del proceso de definición de la arquitectura de los sistemas de gestión de información biomédica en la facultad seis.
- Definición del modelo para el diseño y evaluación de la arquitectura.
- Selección de técnicas y herramientas de soporte al modelo.
- Aplicación del modelo obtenido a proyectos reales.

Los *Métodos de Investigación Científica* a utilizar son los siguientes:

- El *análisis documental* permite realizar un análisis de la información existente siendo utilizado durante el proceso de revisión bibliográfica que se realiza que desde un enfoque histórico-lógico.
- *El análisis y la síntesis*. El *análisis* es un procedimiento mental mediante el cual un todo complejo se descompone en sus diversas partes y cualidades, permite la división mental del todo en sus múltiples relaciones y componentes. La *síntesis*

establece mentalmente la unión entre las partes previamente analizadas y posibilita descubrir las relaciones esenciales y características generales entre ellas, produce sobre la base de los resultados obtenidos previamente en el análisis.

- El método *analítico-sintético* al descomponer el problema de investigación en elementos por separado y profundizar en el estudio de cada uno de ellos, para luego sintetizarlos en la solución de la propuesta.

- La *modelación*, método mediante el cual se crean abstracciones con vistas a explicar la realidad.

La *Novedad Científica* de la investigación se expresa en sus aportes fundamentales que son los siguientes:

- Concepción de un modelo para el proceso de definición la arquitectura en los sistemas de gestión de información biomédica desarrollados por la facultad seis.

- Desarrollo de una técnica de priorización multidensional de requsitos que brinda soporte visual a la toma de decisiones.

La *Significación Práctica* del trabajo es la siguiente:

- Contar con un modelo adaptado a las características de los sistemas de gestión de información biomédica desarrollados en la facultad seis que facilita el proceso de definición de la arquitectura y contribuye al cumplimiento de los atributos de calidad en los sistemas desarrollados desde la definición de su arquitectura.

- Integración de técnicas que posibiliten mayor facilidad en la aplicación del modelo.

La investigación está estructurada de la siguiente forma:

Capítulo 1: Fundamentación teórica, sección en la que se hace un estudio de algunos aspectos fundamentales de la arquitectura de software como su surgimiento, relación con los atributos de calidad, patrones o estilos arquitectónicos, lenguajes de descripción de la arquitectura, métodos basados en la arquitectura de software que permiten su diseño, descripción o evaluación. También se hace un análisis del proceso de definición de la arquitectura de los sistemas de gestión de información biomédica que se desarrollan en la facultad, así como los resultados obtenidos en este.

Capítulo 2: Modelo propuesto, se describe el modelo propuesto a ser utilizado en el proceso de definición de la arquitectura de los sistemas de gestión de información biomédica desarrollados en la facultad seis, esta descripción se realiza a través de la especificación de las actividades presentes en el mismo. Producto a la importancia que tiene para el modelo el uso de las técnicas y herramientas adecuadas en las distintas fases que así lo requieran, en esta sección también se realiza un análisis y selección de las mismas.

Capítulo 3:Evaluación del modelo, donde se realiza una evaluación el modelo propuesto a ser utilizado en el proceso de definición de la arquitectura, dicha evaluación se ha dividido en tres partes fundamentales.

- Valoraciones generales, para analizar algunos aspectos relevantes del modelo, mostrando de esta forma aquellas características que lo fortalecen.
- Comparación con otros métodos basados en la arquitectura, reflejando el nivel de cumplimiento de ciertos indicadores en el modelo con respecto a dichos métodos.
- Aplicación del modelo, donde se describe los resultados que se obtuvieron tras aplicar el modelo en la definición de la arquitectura de dos proyectos de la facultad seis.

1

Fundamentación Teórica

En el presente capítulo se realiza un estudio de algunos aspectos fundamentales de la arquitectura de software como su surgimiento, relación con los atributos de calidad, patrones o estilos arquitectónicos, lenguajes de descripción de la arquitectura, métodos basados en la arquitectura de software que permiten su diseño, descripción o evaluación. También se hace un análisis del proceso de definición de la arquitectura de los sistemas de gestión de información biomédica que se desarrollan en la facultad, así como los resultados obtenidos en este.

1.1 Arquitectura de Software

El número de definiciones existente sobre el término ASW está en el orden de los tres dígitos [1], pero en el año 2000 se publica la versión definitiva de la recomendación IEEE Std 1471 [bes00], homogenizando la nomenclatura de descripción arquitectónica definiendo como ASW "el nivel conceptual más alto de un sistema en su ambiente, la organización fundamental descrita en términos de componentes, la relación entre ellos con el ambiente, los principios que guían su diseño y evolución".

La ASW está es cada día más visible y es de mucha más importancia para el éxito de los proyectos producto al incremento de las tecnologías a utilizar y la evolución en el desarrollo de sistemas que siempre ha tendido a elevar el nivel de abstracción utilizado. Cuando se define la arquitectura de un sistema, el mismo se descompone en un grupo de estructuras relacionadas que trabajan de forma sincronizada posibilitando satisfacer los requisitos definidos, cada una de estas estructuras constituyes componentes y la relación entre estos puede ser vista como conectores.

Por componente se entiende cada una de las partes o unidades de descomposición en las que se subdivide la funcionalidad de un sistema. En un sentido más genérico, puede hacer referencia a cualquier tipo de elemento estructural, es precisamente con este significado con el que habitualmente se le utiliza en la Arquitectura de Software.

El concepto conector proviene con el objetivo de separar las abstracciones relativas a la funcionalidad y a su interacción [Sha96] .De este modo se realiza una clara separación de intereses, que permite ampliar el nivel de abstracción y aumentar la modularidad del sistema. Lo que se propone no es disponer de dos tipos de componentes, sino de distinguir dos elementos diferenciados. Los componentes realizan una tarea sin preocuparse de cómo estos se relacionan con el resto del sistema; por su parte, los conectores son los que se encargan de resolver todas las cuestiones relativas a la comunicación entre los componentes.

La definición de la arquitectura es una de las decisiones más tempranas durante el desarrollo de un sistema, abarca la separación de las responsabilidades, descomposición en módulos, la coordinación y cooperación entre estos para el cumplimiento de las funcionalidades previstas.

Existe un grupo de elementos a tener en cuenta durante la definición de la arquitectura para fortalecer la calidad de la solución arquitectónica propuesta, tales como: patrones o

[1] Definiciones de arquitectura de software: http://www.sei.cmu.edu/architecture/start/community.cfm

estilos arquitectónicos, atributos de calidad, lenguajes de descripción y métodos basados en la arquitectura. En las secciones siguientes de la presente investigación serán descritos estos elementos.

1.2 Atributos de calidad

Los requisitos no funcionales pueden ser divididos en dos grandes grupos, unos los que responden a la pregunta de ¿cómo debe funcionar el sistema después de desarrollado? y los qué responden a ¿cómo debe ser desarrollado el sistema?, los que corresponden al primer grupo son clasificados como atributos de calidad y los del segundo como restricciones de diseño.

Los atributos de calidad, constituyen indicadores que determinan la calidad del sistema desarrollado; de estos a los que más atención debe prestársele durante el desarrollo de los sistemas según la Conferencia Internacional de Ingeniería de Software del año 2002 [Off02] son: Fiabilidad, Usabilidad, Seguridad, Rendimiento, Disponibilidad, Escalabilidad y Mantenibilidad.

- *Rendimiento*: tiempo que requiere el sistema para responder a un evento o estímulo, o bien el número de eventos procesados en un intervalo de tiempo.
- *Seguridad*: medida de la capacidad del sistema para resistir intentos de uso y negación de servicios a usuarios no autorizados sin restar servicios a los usuarios autorizados.
- *Disponibilidad*: proporción del tiempo en el que el sistema está en ejecución, medido muchas veces como el tiempo entre las fallas o la rapidez en que puede reiniciar las operaciones cuando ocurre una falla.
- *Fiabilidad*: habilidad del sistema de hacer la tarea para la cual fue programado.
- *Escalabilidad*: capacidad de un sistema de incrementar su rendimiento de forma proporcional al incremento de hardware o para atender más usuarios concurrentes que el número para el cual fue diseñado.
- *Modificabilidad*: capacidad de hacer cambios en el sistema o agregar funcionalidades de forma rápida y poco costosa.

Tal y como afirman Len Bass, Mark Klein y Felix Bachman [BKB00] "el diseño de la arquitectura de software está estrechamente relacionado con el grado de cumplimiento de los atributos de calidad", a continuación se describe un escenario que confirma el planteamiento anterior:

La separación de las responsabilidades en módulos o subsistemas es una de las decisiones tomadas durante la definición de la arquitectura, una vez ejecutada se podrán hacer un grupo de interrogantes sobre los atributos de calidad: ¿Qué tan modificable es el sistema? ¿En cuánto se afectó el rendimiento? ¿Los nuevos módulos o subsistemas afectan la disponibilidad, seguridad o usabilidad? Cualquier decisión que se tome sobre la estructura del sistema trae implicaciones en el cumplimiento de los atributos de calidad.

A medida que la dimensión y complejidad de los sistemas incrementa, el cumplimiento o no de ciertos atributos de calidad afecta en un mayor grado a otros atributos; los ejemplos más comunes son la modificabilidad, el rendimiento, la seguridad y la disponibilidad, tal y como se muestra en la figura 1.1.

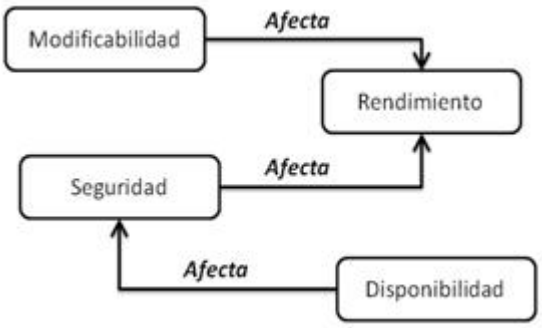


FIGURA 1.1: Dependencia entre atributos de calidad.

A pesar de los atributos de calidad constituir elementos claves para el éxito de los sistemas, en la actualidad no juegan el papel fundamental durante el proceso de definición de la arquitectura [BEL+03](ver figura 1.2), llegando en ocasiones al desarrollo de soluciones que no cumplen con las metas deseadas.

1.3 Patrones o estilos arquitectónicos

En el epígrafe 1.1 se caracterizó la arquitectura de software como la disciplina que aborda los aspectos estructurales de las aplicaciones durante la fase de diseño preliminar o

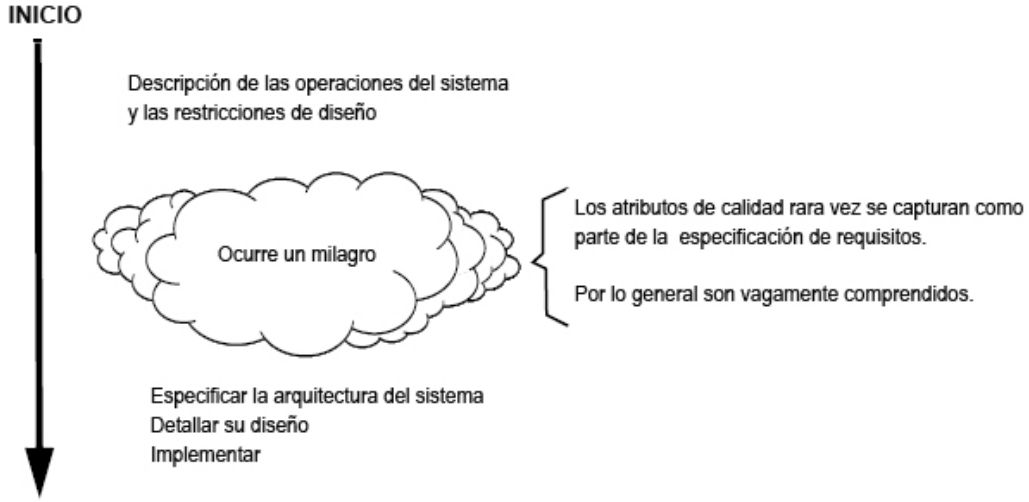


Figura 1.2: Desarrollo tradicional de los sistemas informáticos [BEL+03].

diseño de alto nivel, que tiene por objeto el estudio de la arquitectura de las aplicaciones y la identificación, definición y uso de patrones estructurales para la resolución de las cuestiones que surgen en esta fase del diseño.

Se entiende por patrón como una solución probada que se puede aplicar con éxito a un determinado tipo de problemas que aparecen repetidamente. El establecimiento de estos patrones comunes es lo que posibilita el aprovechamiento de la experiencia acumulada en el diseño de aplicaciones. Un diseñador experimentado producirá diseños más simples, robustos y generales, y más fácilmente adaptables al cambio. Por otra parte, un buen diseño no debe ser específico de una aplicación concreta, sino que debe basarse en soluciones que han funcionado bien en otras ocasiones.

Si el ámbito de aplicación del patrón es de bajo nivel, o diseño detallado, estarían hablando entonces de los llamados patrones de diseño. El uso de un determinado patrón de diseño no afecta a la estructura general de una aplicación, sino sólo a una parte puntual de la misma.

Por el contrario si el ámbito del patrón es de alto nivel, que es precisamente donde se enmarca la arquitectura de software, se estaría hablando de los patrones arquitectónicos, los cuáles son caracterizados por Carlos Canal en su tesis doctoral [Can01] como: “esquemas de organización de los sistemas de software que determinan cuál va a ser la estructura

de los mismos mediante el establecimiento de su división en subsistemas, indicando las responsabilidades de cada uno de estos subsistemas y las reglas y criterios que rigen las relaciones entre ellos". Estos patrones también son conocidos como estilos arquitectónicos.

Los patrones arquitectónicos soportan el desarrollo, el mantenimiento y evolución de sistemas complejos y de gran escala, fundamentados en la reutilización de diseños ya probados; disminuyen los tiempos de desarrollo e incrementan las garantías de éxito.

Cuando se define el uso de un estilo arquitectónico se tiene en cuenta el patrón de organización general, los tipos de componentes presentes habitualmente en el estilo y las interacciones que se establecen entre ellos. Además de determinar los tipos de componentes y conectores involucrados, un estilo arquitectónico indicará cuáles son los patrones y restricciones de interconexión o composición entre ellos. Asociadas a cada estilo hay una serie de propiedades que lo caracterizan, determinando sus ventajas e inconvenientes, que condicionan la elección de uno u otro estilo para resolver un determinado problema, así como la combinación de los mismos.

Entre los principales estilos arquitectónicos se pueden encontrar los siguientes:

- Estilo de flujo de datos.
  - Tuberías y filtros.
- Estilo centrados en datos.
  - Arquitecturas de pizarra.
- Estilo de llamada y retorno.
  - Modelo vista controlador (MVC).
  - Arquitectura en capas.
  - Arquitectura orientada a objetos.
  - Arquitectura basada en componentes.
- Estilo de código móvil.
  - Arquitectura de máquinas virtuales.
- Estilo heterogéneos.
  - Sistemas de control de procesos.
  - Arquitecturas basadas en atributos.
- Estilo peer-to-peer.
  - Arquitecturas basadas en eventos.
  - Arquitecturas orientadas a servicios.
  - Arquitecturas basadas en recursos.

El uso de los patrones arquitectónicos incrementa la productividad, las probabilidades éxito en el diseño arquitectónico y de satisfacción de los atributos de calidad; según Anna Grimán, María Pérez y Luis Mendoza [GPM05] "su impacto en la calidad se encuentra

especialmente relacionado con la organización de los componentes (mantenibilidad) y el manejo de peticiones o eventos (rendimiento, funcionalidad y fiabilidad)".

1.4 Lenguajes de descripción de la arquitectura

Una vez que el arquitecto de software, tras analizar los requerimientos, decide delinear su estrategia y articular los estilos y patrones, debe expresar las características de su sistema, en otras palabras, modelarlo, aplicando una convención gráfica o algún lenguaje avanzado de alto nivel de abstracción; para tal fin son utilizados los lenguajes de descripción de la arquitectura (ADL).

Los ADL se remontan a los lenguajes de interconexión de módulos (MIL) de la década de 1970, pero se comenzaron a desarrollar con su denominación actual a partir de la década de 1990, poco después de fundada la propia arquitectura de software como especialidad profesional. Estos lenguajes surgen por la necesidad de satisfacer los requerimientos descriptivos de alto nivel de abstracción que las herramientas basadas en objeto en general y UML en particular no cumplen satisfactoriamente.

Es real que el modelado orientado a objetos de sistemas basados en componentes posee cierto número de rasgos muy convenientes a la hora de diseñar o al menos describir un sistema. En primer lugar, las notaciones de objeto resultan familiares a un gran número de ingenieros de software. Proporcionan un mapeo directo entre un modelo y una implementación y se dispone de un repertorio nutrido de herramientas tanto comerciales como libres para trabajar con ellas; implementan además métodos bien definidos para desarrollar sistemas a partir de un conjunto de requerimientos.

Pero la descripción de sistemas basados en componentes presenta también limitaciones serias, que no son de detalle sino más bien estructurales. En primer lugar, sólo proporcionan una única forma de interconexión primitiva: la invocación de método [GMW00] . Esto hace difícil modelar formas de interacción más ricas o diferentes. En segundo orden, no soporta la definición de familias de sistemas o estilos; no hay recurso sintáctico alguno para caracterizar clases de sistemas en términos de las restricciones de diseño que debería observar cada miembro de la familia. En última instancia, no brindan soporte formal para caracterizar y analizar propiedades no funcionales, lo que hace difícil razonar sobre propiedades críticas del sistema, tales como rendimiento, modificabilidad y disponibilidad.Existe un grupo de ADL muy usado en la actualidad, los cuáles son resumidos en la tabla 1.4.

TABLA 1.1: Lenguajes de descripción de la arquitectura de software más usados.

Nombre	**Fecha**	**Investigador - Entidad**	**Observaciones**
Acme	1995	Monroe Garlan (CMU), Wile (USC)	Lenguaje de intercambio de ADLs
Aesop	1994	ADL asociado a Acme	ADL de propósito general, énfasis en estilos
C2 SADL	1996	Taylor/Medvidovic	ADL específico de estilo
Darwin	1991	Magee, Dulay,Eisenbach, Kramer	ADL con énfasis en dinámica
Jacal	1997	Kicillof ,Yankelevich (Universidad de Buenos Aires)	ADL - Notación de alto nivel para descripción y prototipado
Rapide	1990	Rapide	ADL & simulación
UniCon	1995	Shaw (CMU)	ADL de propósito general, énfasis en conectores y estilos
Wright	1994	Garlan (CMU)	ADL de propósito general, énfasis en Comunicación
LEDA	2000	Málaga (GISUM)	Lenguaje de especificación y validación de arquitecturas

Utilizando un ADL los arquitectos realizan una o varias modelaciones del sistema, las cuales se consideran vistas arquitectónicas. Las vistas arquitectónicas son representaciones de aspectos globales que tienen significado para uno o más *stakeholders*[2] , se consideran apoyo para los arquitectos principiantes, guía para los activos y soporte para los expertos [RW05]; el número de vistas que debe utilizarse para describir una arquitectura no es finito, este depende de aquellos elementos que son preocupantes o de interés de los stakeholders, lo que se define como punto de vista.

Un punto de vista no es más que una colección de patrones, plantillas o convenciones para la construcción de un tipo de vista en específico; esta define los aspectos preocupantes o de relevancia para uno o varios stakeholders los cuales son reflejados en una vista.

[2]Stakeholders: Cualquier grupo o individuo que pueda afectar o ser afectado por el logro de los objetivos de una organización.

1.5 Métodos basados en la Arquitectura de Software

Definir la arquitectura de un sistema informático es aquella acción mediante la cual este se descompone en componentes y se establecen sus relaciones, creando una estructura de elementos de software capaz de satisfacer los requisitos del sistema. Las etapas que deben estar presentes durante la definición de la arquitectura son:

1. Comprender el problema a resolver.
2. Identificar los elementos de diseño y sus relaciones.
3. Evaluar el diseño de la arquitectura.
4. Aplicar el diseño propuesto.

Las tres fuentes principales que posibilitan definir una arquitectura son la reutilización, la intuición y los métodos [FCK07], las cuales pueden ser descrita de la siguiente manera:

- *La reutilización*: Cuando ya se ha probado una arquitectura bajo ciertas circunstancias y se han obtenido resultados satisfactorios, utilizarla nuevamente bajo condiciones iguales o similares. Ejemplo de esto pueden citarse la arquitectura de dominio específico y la arquitectura de líneas de productos.
- *La intuición*: Improvisación de los arquitectos de software basados en sus experiencias. Por esta vía se desarrollan diseños arquitectónicos a riesgos, sólo aquellos arquitectos con vasta experiencia deberían utilizar esta variante.
- *Los métodos*: Procedimientos y técnicas que posibilitan bajo ciertas circunstancias describir, evaluar o diseñar la arquitectura de los sistemas centrados en el análisis de los requisitos.

Los métodos constituyen la vía más adecuada para aquellas organizaciones que no cuentan con elevada experiencia en el desarrollo de software, disminuyen las probabilidades de dar soluciones arquitectónicas fallidas, sistematizan el proceso de definición de la arquitectura y posibilitan incrementar las competencias arquitectónicas de los equipos de desarrollo con mayor rapidez.

Desde que el campo de la Arquitectura de Software ha cobrado un elevado nivel de importancia, investigadores han definido un grupo de métodos enfocados en el diseño, la descripción o evaluación, dentro de estos se pueden encontrar los siguientes:

Método de Análisis de Arquitecturas de Software (SAAM) es un método de evaluación de la arquitectura basado en escenarios que posibilita a su vez realizar comparaciones entres propuestas arquitectónicas con el objetivo de seleccionar la más adecuada a las necesidades de la organización. Según sus creadores Kazman, Bass y Webb [KBWA94] "...su propósito fundamental no es criticar o elogiar cierta arquitectura, sino proporcionar la herramienta necesaria para determinar si dicha arquitectura cumple con las necesidades de la organización". Este método está enfocado principalmente en los atributos de calidad Modificabilidad y funcionabilidad.

Architecture Tradeoff Analysis Method (ATAM) es la evolución de SAAM, para el análisis de arquitecturas software basado en la utilización de escenarios, en el que la evaluación de una arquitectura no se realiza para identificar de manera precisa el comportamiento de un atributo de calidad, lo cual no es posible en fases tempranas del diseño por falta de información, sino para descubrir qué decisiones de diseño afectan de una u otra forma a los atributos de calidad. Esto se hace a través de la identificación de:

- Riesgos: decisiones aplazadas o decisiones cuyo efecto no se alcanza a valorar.
- Puntos sensibles: partes de la arquitectura que pueden tener mucha influencia en algún atributo de calidad.
- Puntos de compromiso: partes de la arquitectura cuya modificación significa mejorar algún atributo de calidad a costa de empeorar otro.

El objetivo es poder tomar decisiones razonadas acerca del diseño para, en sucesivos análisis, dedicar más esfuerzo a completar esas partes de la arquitectura. ATAM es un método de análisis organizado en torno a la idea de que los estilos arquitectónicos son los principales determinantes de los atributos de calidad arquitectónica[KKC00].

Quality Attribute Workshop (QAW) este es un método complementario, cuyo objetivo principal es identificar, priorizar y describir los escenarios de atributos de calidad antes de proceder a definir la arquitectura, posibilitando de esta forma un proceso de evaluación de la arquitectura mucho más certero. Se centra en los aspectos arquitectónicos significativos e involucra tanto a los desarrolladores como a los clientes [BEL$^+$03].

Attribute-Driven Design (ADD) es un método de diseño arquitectónico dirigido por los atributos de calidad que se quiere que posea el sistema, más que por la funcionalidad de la aplicación, que queda en un segundo nivel. Las entradas para el método ADD son los requisitos funcionales, restricciones del diseño y los atributos de calidad expresados en forma de escenarios generales, y la salida es una propuesta arquitectónica, la cual

cumple en un alto grado con los atributos de calidad deseados y proporciona la estructura necesaria para dotar al sistema de la funcionalidad requerida.

El Proceso Unificado de Desarrollo de Software (RUP por sus siglas en inglés) es un proceso de desarrollo de software desarrollado y comercializado por Rational Software. El método de diseño de la rquitectura incluido por esta metodología se vasa en las 4+1 vistas arquitectónicas; cuatro vistas para describir el diseño: vista lógica, vista de proceso, vista lógica y vista de despliegue, usando una vista de casos de uso para relacionarse el diseño con el contexto y los objetivos. En RUP la arquitectura se va definiendo e varias iteraciones, en las que se profundiza en la descripción de la arquitectura a través de las 4+1 vistas. Este proceso toma como base fundamental los casos de usos arquitectónicamente significativos.

Siemens Four-Views (S4V) es un método desarrollado por investigadores de la empresa Siemens, se basa fundamentalmente en separar en cuatro vistas (conceptual, modular, ejecución y código) las "percepciones"del sistema en aras de reducir la complejidad de diseñar la arquitectura. Estas vistas se desarrollan a través de un análisis global realizado a partir de las restricciones de diseño, requisitos y atributos de calidad del sistema.

Active Reviews for Intermediate Designs (ARID) es un método de análisis de arquitecturas resultado de combinar Active Design Reviews (ADR) y ATAM. Resulta más ligero que ATAM y es útil para ser aplicado en etapas más tempranas del diseño arquitectónico e incluso, sobre partes del sistema en lugar de sobre el sistema completo. Paul C. Clements [Cle00] afirma que "...permite realizar un análisis de diseño más detallado que el ATAM". La evaluación se basa en detallar el funcionamiento del sistema en una serie de escenarios predefinidos, a través de lo cual se pueden identificar posibles puntos problemáticos de la arquitectura.

Cost-Benefit Analysis Method (CBAM) es un método para evaluar los beneficios, costes y riesgos de las diferentes decisiones que se toman para diseñar la arquitectura software del sistema. Al igual que QAW, también está pensado para su integración con ATAM, ya que los resultados producidos por la evaluación de la arquitectura, especialmente las estrategias arquitectónicas a seguir, se utilizan como entrada en CBAM para tomar decisiones basadas en criterios económicos.

Software Architecture Comparison Method (SACAM) es una técnica usada para comparar arquitecturas candidatas, fue desarrollada como apoyo técnico en el contexto de ayudar a las organizaciones en la selección de las arquitecturas candidatas para las nuevas líneas de productos. Sin embargo puede ser utilizada también para analizar varias propuestas de arquitecturas para una solución informática.

Views and Beyond (V&B) es la propuesta realizada en el Instituto de Ingeniería de Software (SEI por sus siglas en inglés) para documentar la arquitectura software de un sistema [CGB+02]. De acuerdo con la definición de arquitectura como la estructura o estructuras del sistema, V&B propone la definición de una serie de vistas relevantes de la arquitectura software del sistema, documentando cada una de ellas, así como las características que afecten a más de una o a todas en general. El número y el tipo de las vistas de un sistema no están determinados a priori, aunque en se pueden agrupar en vistas de módulos, vistas de componentes y conectores, vistas de localización y combinaciones de ellas. Para documentar las vistas de una manera sistemática y homogénea han creado unas plantillas que contienen la estructura de la información que debe aportarse sobre cada vista.

El SEI es un instituto federal estadounidense de investigación y desarrollo, fundado por el Congreso de los Estados Unidos en 1984 para desarrollar modelos de evaluación y mejora en el desarrollo de software. Financiado por el Departamento de Defensa de los Estados Unidos y administrado por la Universidad Carnegie Mellon.

En los últimos años, el SEI ha dedicado una de sus líneas de investigación a los aspectos relacionados con la arquitectura de software convirtiéndose en la organización con mayores resultados científicos en ese campo. De estas investigaciones ha emergido un proceso enfocado al diseño y evaluación de las arquitecturas de software que se enmarca en un contexto organizacional. Dicho proceso se muestra en la figura 1.3; el cual se centra en algunos de los métodos anteriormente descritos, estos son:

1. QAW: Con el objetivo de identificar, describir y priorizar los atributos de calidad presentes en el problema a resolver.
2. ADD: A través de tácticas y patrones diseñar la arquitectura del sistema tomando como base la descripción y priorización de los atributos de calidad, así como las funcionalidades arquitectónicamente significativas y restricciones de diseño.
3. V&B: Para describir la arquitectura diseñada a través de vistas arquitectónicas.
4. ATAM: Para realizar una evaluación de la arquitectura definida.

1.6 Sistemas de gestión de información biomédica

En la era que se vive hoy día, una era de la información, uno de los principales problemas a los que se enfrenta la humanidad es precisamente el exceso de dicha información y el

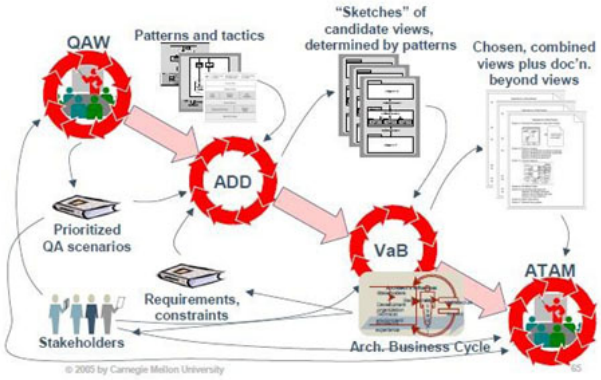


Figura 1.3: Proceso de diseño y evaluación de la arquitectura propuesto por el SEI [Cer09]

cómo manejarla. La gestión de la información ocupa, cada vez más, un espacio mayor en la economía de los países a escala mundial [Q+02]; los sistemas de gestión de información son aquellas herramientas que posibilitan dicho proceso.

Por otra parte el acercamiento de la Informática Médica y la Bioinformática ha dado lugar al surgimiento de un nuevo tipo de aproximación, denominada Informática Biomédica o Bioinformática Médica [VL07] tal y como se muesrtra en la figura 1.4.

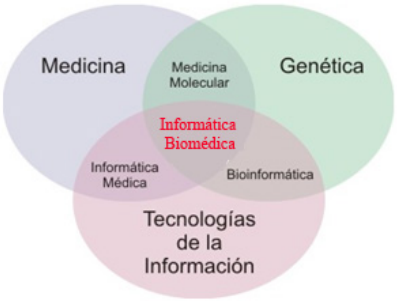


Figura 1.4: Informática Biomédica [VL07]

La Informática Biomédica, en su afán por comprender un gran número de fenómenos naturales, con el objetivo de entender su funcionamiento y proponer soluciones a enfermedades, epidemias o virus ha adquirido con el tiempo un elevado cumulo de información que es vital para el éxito de las investigaciones; los sistemas de gestión de información biomédica, posibilitan el control, seguimiento y análisis de dicha información.

En la facultad seis de la Universidad de las Ciencias Informáticas se encuentra en desarrollo un grupo de proyectos de gestión de información biomédica con alta importancia para el Polo Científico de la capital cubana, ejemplo de estos son el sistema informático de genética médica con el Centro Nacional de Genética Médica, sistema de ayuda médica para la atención a la Dislipoproteinemias con el hospital Hermanos Ameijeiras, herramienta para la automatización de los estudios clínico genéticos que hoy se han llevado a cabo en cuatro países del continente americano.

Los recursos humanos con los que se cuenta para el desarrollo de los sistemas carecen de experiencia en el campo de la producción de software, un alto por ciento de estos recursos no cuentan con una elevada preparación en las tecnologías empleadas; a pesar de que en estos años se ha ido adquiriendo cierta madurez todavía quedan aspectos por limar, lograr un papel más activo de los clientes, aumentar la calidad de las soluciones, incrementar las competencias del personal son algunos ejemplos.

En cuanto a la calidad de las soluciones existe un factor que ha influido notablemente en esto y que a nivel mundial ha afectado a otras empresas de la industria informática, enfocarse solamente en los requisitos funcionales a la hora de diseñar las arquitecturas de los sistemas y no tener presentes los atributos de calidad. Lo anterior trae consigo un grupo de inconvenientes como son:

- Reutilización de conocimiento y fragmentos de código en lugar de funcionalidades o componentes lo que permitiría aumentar notablemente la productividad.
- Utilizar parches en etapas tardías del sistema disminuyendo el rendimiento y la modularidad.
- Carencia de interoperabilidad entre las soluciones desarrolladas.

Otra problemática que se han detectado en el proceso de desarrollo de software es que los equipos de trabajo presentan dificultad para:

- Capturar y documentar los atributos de calidad.
- Realizar un diseño sistemático de la arquitectura.
- Realizar una documentación adecuada de la arquitectura.
- Evaluar el diseño arquitectónico.
- Comparar y discutir alternativas.

Elementos que afirman la existencia de una baja competencia arquitectural. Este tema fue abordado en la novena Conferencia de Arquitectura de Software auspiciada por el Instituto de Ingenieros Eléctrico y Electrónico (IEEE por sus siglas en inglés) y la Federación Internacional para el Procesamiento de la Información (IFIP por sus siglas en inglés) titulada "Una conferencia para y por Arquitectos de Software"; en la que un grupo de investigadores del SEI afirman [CKK+07]"...equipar los arquitectos con métodos y herramientas, elevar sus competencias y contar con el apoyo de la organización son los tres elementos que se deben tener en cuenta para incrementar el nivel de competencia arquitectural".

Dentro de la estrategia de formación se contemplan cursos y talleres de arquitectura de software en aras de incrementar las competencias de los equipos de trabajo, previendo aumentar la calidad de los productos desarrollados. Sin embargo aún no se han definido procedimientos para la definición, evaluación o descripción de la arquitectura de los sistemas desarrollado.

Por las características que presenta la facultad y haciendo un análisis de los temas tratados en el epígrafe 1.5 se puede deducir que la el empleo de los métodos basados en la arquitectura podría contribuir a solucionar la problemática planteada en esta investigación.

Tras hacer un análisis de los métodos basados en la arquitectura, respecto a la posible aplicación de estos en el proceso de definición de la arquitectura se detectaron un grupo de desventajas:

- Son genéricos, están enfocados a cualquier tipo de proyectos, desde el desarrollo de componentes embebidos en soluciones de hardware hasta sistemas informáticos que manejan gran volumen de información como es el caso de los ERP.

- Se especializan en un área determinada del proceso de definición de la arquitectura (diseño, evaluación o descripción).

- Describen qué actividades desarrollar en cada caso pero no se define el cómo hacerlas, aspecto clave a tener en cuenta cuando el equipo de desarrollo no tiene un elevado nivel de competencias en el tema.

- Presuponen la existencia de elevadas competencias dentro de la organización en el tema de la arquitectura de software y un alto grado de colaboración e implicación de los clientes en el diseño de esta.

1.7 Conclusiones parciales

- La arquitectura de software constituye un elemento clave para alcanzar la calidad deseada en el desarrollo de sistemas informáticos y presenta cada día mucha más importancia producto a la constante evolución de dicho campo de la ingeniería de software.
- El uso de los métodos en el proceso de definición de la arquitectura constituye una vía sistémica a través de la cual los arquitectos sin altas competencias arquitecturales pueden encontrar soluciones, garantizando desde etapas tempranas del desarrollo elevadas probabilidades de éxito en los sistemas.
- El proceso de definición de la arquitectura de los sistemas desarrollados en la facultad seis carece de buenas prácticas arquitectónicas que le impiden desarrollar soluciones con alto grado de satisfacción de los atributos de calidad.

De lo anterior se infiere la necesidad de un modelo para el diseño y evaluación de la arquitectura de los sistemas desarrollados en la facultad seis que provea las herramientas necesarias para incrementar la calidad de las soluciones desde el proceso de definición de su arquitectura.

2

Modelo propuesto

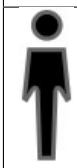
En el presente capítulo se describe el modelo propuesto a ser utilizado en el proceso de definición de la arquitectura de los sistemas de gestión de información biomédica desarrollados en la facultad seis, esta descripción se realiza a través de la especificación de las actividades presentes en el mismo. Producto a la importancia que tiene para el modelo el uso de las técnicas y herramientas adecuadas en las distintas fases que así lo requieran, en esta sección también se realiza un análisis y selección de las mismas.

2.1 Descripción del modelo

El presente modelo tiene como *objetivo* proporcionar un conjunto de actividades, técnicas y herramientas a los arquitectos que les permita diseñar y evaluar la arquitectura de los proyectos de una forma más ordenada y sistemática que los métodos usados en la actualidad centrados en los requisitos funcionales y en la experiencia.

En la tabla 2.1 se describen los actores que intervienen en el proceso de deficinicón de la arquitectura según el modelo propuesto.

TABLA 2.1: Actores que intervienen en el proceso de definición de la arquitectura.

Estereotipo	**Rol**	**Descripción**
	Arquitecto	Miembro del equipo de desarrollo que tienen como misión fundamental definir las estructuras arquitectónicas de los distintos sistemas y dar seguimiento a su construcción.
	Cliente	Representantes o usuarios finales de los sistemas que se desarrollen.
	Responsable de sistemas externos	Responsable del desarrollo o mantenimiento de sistemas externos que posean funcionalidades o componentes que puedan ser reutilizados en el desarrollo de las nuevas aplicaciones.

El modelo toma como *entrada* la descripción de los requisitos del sistema a desarrollar donde deben estar recogidos y correctamente documentados los atributos de calidad y las restricciones de diseño.

El conjunto de *actividades* comprendidas son:

1. Determinar si los requisitos actuales son suficientes para definir la arquitectura.

2. Definir los componentes externos que serán utilizados.
3. Seleccionar un elemento a descomponer del sistema.
4. Identificar los conductores arquitectónicos.
5. Seleccionar un concepto de diseño que satisfaga los conductores arquitectónicos.
6. Asignar responsabilidades y definir interfaces.
7. Describir la arquitectura del sistema.
8. Verificar el cumplimiento de los requisitos.

Una vez ejecutada las actividades se obtiene como *salida* la descripción de la arquitectura del sistema definida, basada en el cumplimiento de los requisitos del mismo. En la figura 2.1 de la página 20 se puede apreciar una representación gráfica del modelo.

2.1.1 Determinar si los requisitos actuales son suficientes para definir la arquitectura

Objetivos: Verificar que se encuentran suficientes requisitos para proceder a definir la arquitectura, priorizar los requisitos desde el punto de vista del cliente.

Actores: Arquitecto, Cliente.

Artefactos: Lista de requisitos.

Técnicas utilizadas: Priorización de requisitos a través del método de los 100 puntos. Tormenta de ideas.

Entrada: Listado de los requisitos del sistema.

Salida: Listado de los requisitos arquitectónicamente significativos priorizados por parte de los clientes.

Debido a que el modelo propuesto centra el proceso de definición de la arquitectura en los requisitos del sistema, antes de dar los primeros pasos es necesario verificar que los requisitos con los que se cuenta son suficientes para proceder.

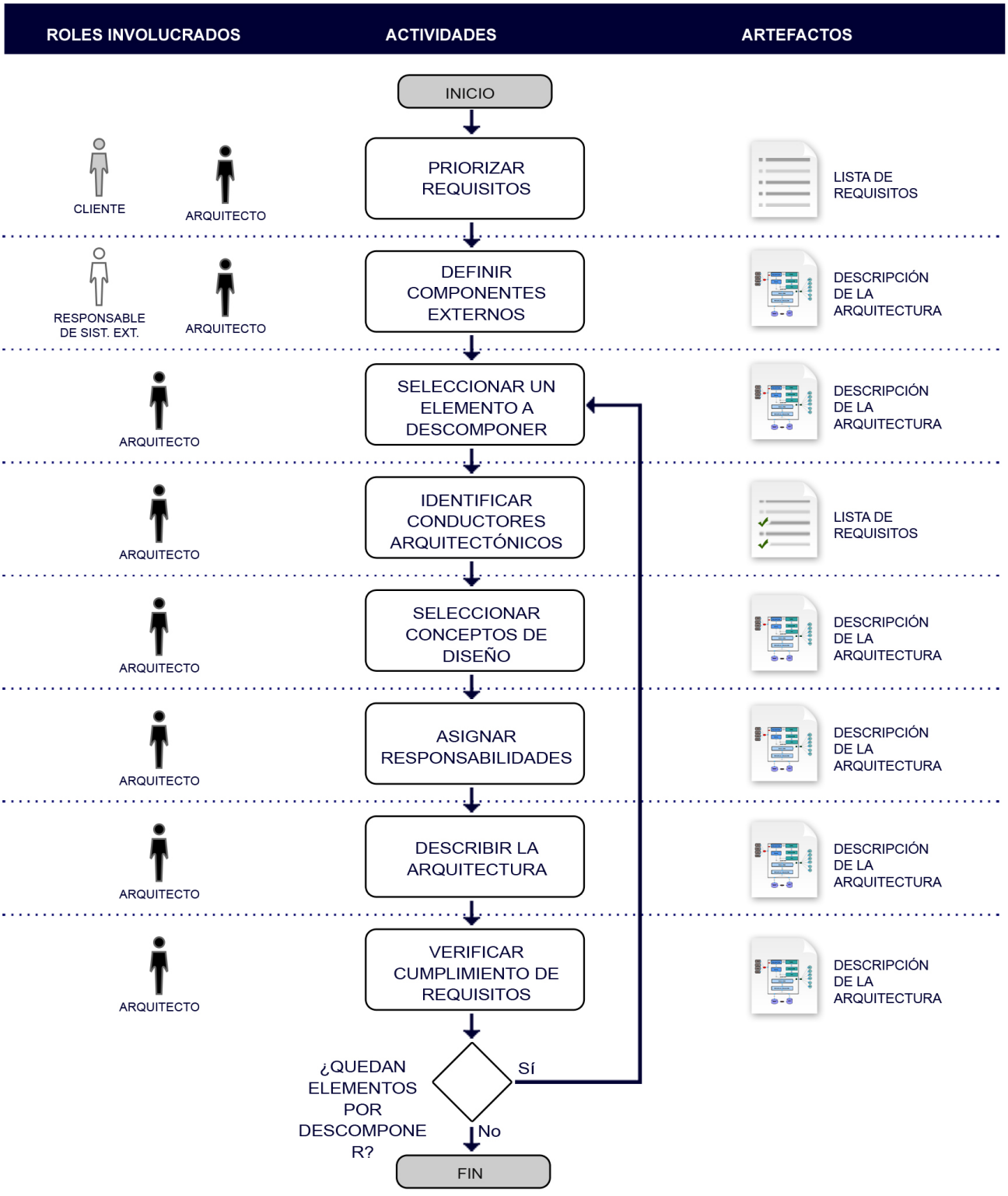


FIGURA 2.1: Representación gráfica del modelo.

Seleccionar los requisitos a tener en cuenta es un aspecto clave y de gran prioridad para el éxito en la construcción de la estructura del sistema. Si se incluyen demasiados requisitos se pierde la propiedad de configuración mínima necesaria (o más simple posible) que permite demostrar en fechas tempranas la corrección del diseño arquitectónico en caso de ser necesario; si por el contrario no se incluyen todos los necesarios, probablemente se ignoren requisitos, riesgos o interacciones críticas, lo que impide dar garantía sobre el éxito del proyecto antes de realizar los mayores esfuerzos y gastos en la fase de producción [CK03].

Esta primera etapa es sumamente importante, hasta el año 2006 más del 60 % de los errores cometidos durante la definición de arquitecturas estaba dado por la omisión de requisitos durante el análisis de la solución a desarrollar [RR06].

Para lograr los objetivos de esta etapa se propone hacer uso de tormentas de ideas en un taller donde se encuentre la mayor cantidad de clientes posibles, con el objetivo de hacer una revisión para cerciorarse que no falten requisitos; la tormenta de ideas es una técnica efectiva para la elicitación de requisitos especialmente cuando los clientes no tienen una vasta experiencia en la participación en proyectos de desarrollo de software [TT05]. Los miembros del equipo de desarrollo que se encuentren presente juegan un papel pasivo en dicho taller, el objetivo de su presencia en el mismo es dar asesoramiento técnico a los clientes en caso de ser necesario, fungir como moderador del debate y garantizar que sólo los requisitos que influyen sobre la arquitectura (aquellos que tienen impacto sobre la estructura del sistema) sean los seleccionados para los siguientes pasos del modelo.

Para facilitar la evaluación del cumplimiento de los atributos de calidad es necesario describir el escenario de aquellos requisitos ubicados en ese grupo teniendo en cuenta los siguientes aspectos: fuente del estímulo, estímulo (es el evento de entrada), ambiente, artefacto, respuesta al estímulo, medida de la respuesta [VS08] ; puede utilizarse como material de apoyo el manual “Guía de participantes en el taller de atributos de calidad”, propuesto por el grupo de investigación de la arquitectura de software del SEI.

Una vez que se tienen todos los requisitos se procede a priorizarlos desde el punto de vista del cliente, a través del método de los 100 puntos. Se le da la posibilidad a cada cliente presente en el taller de distribuir 100 puntos entre todos los requisitos que fueron detectados dándole mayor puntuación a los más importantes teniendo en cuenta su criterio personal.

Una vez realizada la votación se procede a determinar el nivel de prioridad de cada requisito a través de la matriz de cálculo de prioridad mostrada en la tabla 2.2, siendo P_{ij} la puntuación dada por el cliente i al requisito j y PF_{Rj} el promedio de las puntuaciones dadas al requisito j.

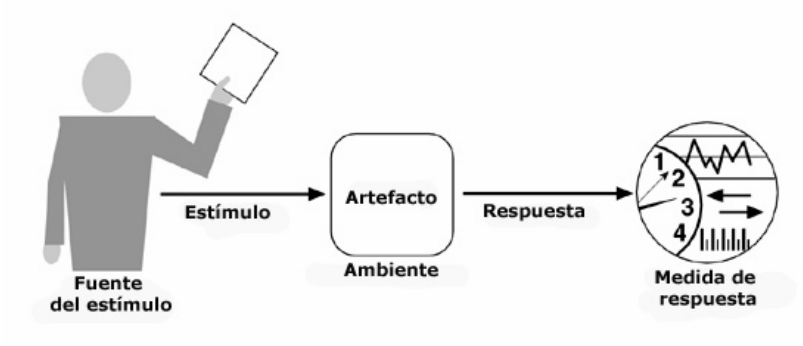


FIGURA 2.2: Escenarios de atributos de calidad.

TABLA 2.2: Matriz de cálculo de la prioridad de requisitos.

	$Requisito_1$	$Requisito_2$	$Requisito_3$	...	$Requisito_n$
$Cliente_1$	P_{11}	P_{12}	P_{13}	...	P_{1n}
$Cliente_2$	P_{21}	P_{22}	P_{23}	...	P_{2n}
$Cliente_3$	P_{31}	P_{32}	P_{33}	...	P_{3n}
...	...	...	...	...	...
$Cliente_n$	P_{n1}	P_{n2}	P_{n3}	...	P_{nn}
$Prioridad$	PF_{R1}	PF_{R2}	PF_{R3}	...	PF_{Rn}

2.1.2 Definir componentes externos que serán utilizados

Objetivo: Identificar aquellos componentes externos que puedan utilizarse en el desarrollo del sistema.

Actores: Arquitecto, Responsable de sistema externo.

Artefactos: Documento de descripción de la arquitectura..

Entrada: Descripción de los requisitos del sistema.

Salida: Componentes o funcionalidades a utilizar durante la definición de la arquitectura del sistema.

Esta actividad está dedicada a la identificación de los sistemas externos que se convertirán en fuente o destino de la información del sistema.

A partir del catálogo de las soluciones ya desarrolladas por la facultad o presentes en el dominio de aplicación del sistema a desarrollar, se hace un estudio de aquellos componentes o funcionalidades que se puedan reutilizar.

Para cada uno de los componentes a utilizar debe definirse la forma en que se interconectará con el sistema, interfaz de comunicación a utilizar, requisito(s) involucrados, parámetros involucrados en la comunicación tanto de la petición y como de la respuesta.

2.1.3 Seleccionar un elemento a descomponer del sistema

Objetivo: Seleccionar aquel elemento del sistema que se procederá a descomponer.

Actores: Arquitecto.

Artefactos: Documento de descripción de la arquitectura.

Entrada: Descripción actual de la arquitectura.

Salida: La selección del elemento del sistema que será descompuesto en las siguientes actividades.

En esta actividad se selecciona un elemento a descomponer del sistema. Si es la primera iteración se selecciona el sistema en su conjunto, pero en las siguientes iteraciones se seleccionaría algún elemento del sistema que formaría parte de la estructura general definida en alguna de las iteraciones anteriores.

En caso de no ser la primera iteración, para seleccionar el elemento puede tenerse en cuenta alguna de las siguientes áreas de interés:

Basado en el conocimiento actual de la arquitectura y criterios tales como:

- El grado de dependencia que exista entre los requisitos que aún no han sido instanciados con el nuevo elemento a seleccionar.
- La estructura que se ha formado con las iteraciones anteriores y la visión futura que se tiene del sistema basado en experiencias previas.

Basado en los riesgos o dificultad que exista para desarrollar el elemento seleccionado:

- El grado de dificultad de satisfacer los requisitos asociados al elemento.
- La familiarización que tenga el equipo de desarrollo con cumplir los requisitos asociados al elemento.

Basado en criterios de la organización de rige el desarrollo del sistema:

- Priorización de alguna tecnología por motivos tanto del cliente como del equipo de desarrollo.
- Reutilización de funcionalidades ya implementadas en sistemas anteriores y/o externos.

2.1.4 Identificar los conductores arquitectónicos

Objetivos: Identificar aquellos requisitos que guiarán la descomposición del elemento seleccionado.

Actores: Arquitecto.

Artefactos: Lista de requisitos.

Técnicas utilizadas: Técnica de priorización de requisitos - Árbol binario de búsqueda.

Entrada: El listado de aquellos requisitos asociados al elemento del sistema que se procederá a descomponer.

Salida: El listado de aquellos requisitos que forman parte de los conductores arquitectónicos en la presente iteración.

Hasta este momento se seleccionó un elemento del sistema a descomponer y han sido priorizados todos los requisitos por parte del cliente. Corresponde ahora identificar todos los requisitos relacionados con el elemento seleccionado en el paso anterior y elegir cuáles de ellos pasarán a formar parte de los conductores arquitectónicos en esta iteración.

El primer aspecto a tener en cuenta en esta actividad es seleccionar cuáles requisitos están relacionados con el elemento que se procederá a descomponer, esto significa identificar de los requisitos cuáles se verán afectados o se le dará cumplimiento con la descomposición del elemento seleccionado.

Una vez seleccionados los requisitos involucrados con el elemento ha llegado el momento de darle la prioridad desde el punto de vista arquitectónico, usando la técnica de priorización basada en árboles binarios de búsqueda.

Luego de ejecutada la técnica anterior se obtendrá un árbol binario con los requisitos, dicho árbol se recorre en pre orden para listar todos los requisitos en orden descendente de prioridad arquitectónica.

Posteriormente se seleccionan los conductores arquitectónicos mediante el "método visual para la selección de los conductores arquitectónico"[1] .

2.1.5 Seleccionar un concepto de diseño que satisfaga los conductores arquitectónicos

Objetivos: Seleccionar aquel concepto de diseño que satisfaga en el mayor grado posible los conductores arquitectónicos para utilizarlo en la descomposición del elemento seleccionado.

Actores: Arquitecto.

Artefactos: Documento de descripción de la arquitectura.

Técnicas utilizadas: Tormenta de ideas.

Entrada: Descripción actual de la arquitectura y los conductores arquitectónicos de la iteración actual.

Salida: Selección de los patrones arquitectónicos a utilizar en la definición de la arquitectura del sistema.

Esta sección tiene como objetivo seleccionar un concepto de diseño (estilo o patrón) a ser aplicado en la descomposición del elemento acorde con los conductores arquitectónicos identificados en el paso anterior, se logrará a través de los siguientes pasos:

- Identificar todas las restricciones de diseño asociada a cada uno de los conductores arquitectónicos.

[1]Método desarrollado en el presente trabajo que será descrito en el epígrafe 2.2.3.

- Seleccionar por cada uno de los conductores arquitectónicos los posibles patrones a utilizar teniendo en cuenta la experiencia en trabajos previos, la estructura actual del sistema, los atributos de calidad asociados o la búsqueda de información en libros especializados.

- Hacer un análisis de los posibles patrones a utilizar y seleccionar el o los más apropiados apoyándose en la matriz de evaluación de los patrones mostrada en la tabla 2.3 ubicando por cada patrón los pros y contras.

- Mediante una tormenta de ideas identifique los pros y los contra que traería consigo poner en práctica cada patrón, en base a un análisis de la matriz seleccione el o los patrones a utilizar en la descomposición del componente.

TABLA 2.3: Matriz de evaluación de los patrones candidatos.

	Patrón 1		Patrón 2		...	Patrón n	
	Pro	Contra	Pro	Contra	...	Pro	Contra
Conductor Arquitectónico 1					...		
Conductor Arquitectónico 2					...		
Conductor Arquitectónico 3					...		
...	...	...	...	...	...	...	...
Conductor Arquitectónico n					...		

2.1.6 Asignar responsabilidades y definir interfaces

Objetivos: Asignar las responsabilidades a los nuevos componentes del sistema. Definir las interfaces de comunicación para dichos componentes.

Actores: Arquitecto.

Artefactos: Documento de descripción de la arquitectura.

Entrada: Los patrones arquitectónicos a ser usados esta iteración y el documento de descripción de la arquitectura.

Salida: La descripción de las responsabilidades de cada uno de los nuevos componentes que surgen en el sistema a partir de la utilización de los patrones arquitectónicos seleccionados en el paso previo.

Teniendo en cuenta los patrones seleccionados en el paso previo definir los nuevos componentes que formarán parte de la arquitectura del sistema, para cada uno de ellos documente las responsabilidades que tendrá sobre el sistema.

Definir para cada uno de los nuevos elementos del diseño cuáles serán sus interfaces con los restantes elementos, comprendiéndose como interfaces la vía a través de la cual obtendrá o facilitará información requerida a componentes externos para cumplir las responsabilidades asignadas en el paso previo.

2.1.7 Describir la arquitectura del sistema

Objetivos: Describir la estructura del sistema a través de vistas arquitectónicas.

Actores: Arquitecto.

Artefactos: Documento de descripción de la arquitectura.

Entrada: Los nuevos componentes que formarán parte de la arquitectura y sus responsabilidades.

Salida: La descripción de la arquitectura a través de las vistas arquitectónicas en la que se incluyen los nuevos componentes.

La descripción de la arquitectura es la descripción formal de la información del sistema, organizada de cierto modo que posibilita el razonamiento sobre las partes estructurales del mismo, esta se lleva a cabo a través de las vistas arquitectónicas.

Tal y como se destacó en el capítulo anterior las vistas que se usen para describir el sistema dependen de aquellos elementos que los stakeholders quieran resaltar. Para los proyectos desarrollados en la facultad, hay dos vistas principales que no deben faltar en la descripción de la arquitectura, la vista de alto nivel de abstracción y la vista de seguridad; en dependencia de la particularidad de cada proyecto pueden adicionarse aquellas vistas que se estimen necesaria para una mejor comprensión de solución arquitectónica elaborada.

La *vista de alto nivel* de abstracción posibilita describir la arquitectura organizacional del sistema en términos de componentes, relaciones y restricciones. Para el desarrollo de esta vista se aconseja hacer uso de un lenguaje de descripción de la arquitectura (ADL) en lugar de UML, tiene como objetivo fundamental mostrar no sólo al equipo de desarrollo

sino también a los clientes cómo quedará estructurada la solución informática.

La seguridad en el procesamiento e intercambio de los datos en aplicaciones es un aspecto al que se le debe prestar especial atención, requiere técnicas y medidas para enfrentarlo. Existen algunas técnicas, una de las más usadas en el área del desarrollo de software para la salud es la denominada técnicas o tecnologías de aumento de la privacidad (PET por sus siglas en inglés) donde se proponen un grupo de aspectos a tener en cuenta para lograr aplicaciones segura tales como encriptación de los datos, accesos limitados, no registrar información privada o destrucción de la misma inmediatamente de ser usada, uso de protocolos seguros, entre otros. La Vista de seguridad tiene como objetivo describir aquellos componentes que contribuyen a la seguridad del sistema así como la relación entre estos.

2.1.8 Verificar el cumplimiento de los requisitos

Objetivos: Verificar el cumplimiento de los requisitos. Realizar un análisis del grado de satisfacción de los atributos de calidad.

Actores: Arquitecto.

Artefactos: Documento de descripción de la arquitectura. Lista de requisitos.

Entrada: Descripción actual de la arquitectura del sistema y sus requisitos.

Salida: Identificación de los puntos sensibles y de compromiso de la arquitectura definida hasta el momento.

La evaluación de la arquitectura consiste en determinar si esta ha cumplido en el grado deseado con los atributos de calidad [BZJ04], la misma naturaleza del modelo propuesto permite ir diseñando la arquitectura centrados en la satisfacción de estos atributos, por lo cual empíricamente mientras se vaya iterando en el diseño de la arquitectura se está evaluando la misma.

De igual forma, para facilitar las decisiones de diseño en iteraciones posteriores es necesario hacer un análisis de cumplimiento de cada atributo de calidad acorde al estado actual de la definición de la arquitectura; otro resultado de gran impacto que se produce con este análisis es la posibilidad de identificar los puntos sensibles y de compromiso.

Los puntos sensibles son aquellos elementos del sistema que si son modificados pueden

afectar el cumplimiento de un atributo de calidad en específico y los puntos de compromiso por su parte son aquellos elementos que si se modifican pueden mejorar algún atributo a costa de empeorar otro.

Cuando se complete esta actividad el sistema estará formado por varios componentes; cada uno de ellos tendrá asociado un grupo de funcionalidades, mientras alguno de estos pueda ser descompuesto se repite la actividad 3 tomando dicho componente como punto de partida, en caso contrario ya estará definida la arquitectura del sistema.

2.2 Técnicas y herramientas de soporte al modelo

El modelo se centra en la descripción y evaluación de la arquitectura a partir de los requisitos del sistema. Hacer la priorización correcta tanto de los requisitos en cada una de las iteraciones y elegir un lenguaje adecuado con el objetivo de describir la arquitectura de una forma comprensible para el equipo de desarrollo y los clientes, son factores claves para el éxito.

En las secciones siguientes se realizará la selección del ADL a utilizar en el modelo y se describirán las técnicas propuestas para priorizar los requisitos en las distintas etapas de modelo que se requiera.

2.2.1 Selección del lenguaje de descripción de la arquitectura

Existen un grupo de alternativas de ADL que pueden ser usadas. Seleccionar una alternativa es el objetivo del presente epígrafe y se realizará a través del proceso de jerarquía analítica (AHP por sus siglas en inglés), desarrollada por Tom Saaty, técnica flexible de análisis de decisión multicriterio, para ayudar a la toma de decisiones [LD05].

Para la aplicación de AHP se deben seguir los siguientes pasos:

1. Identificación de las alternativas.
2. Identificación de los criterios.
3. Ponderación de los criterios.
4. Calcular el score de cada alternativa.

Las alternativas en este caso lo constituyen los distintos ADL descritos anteriormente.

A continuación se procederá a la *identificación de los criterios de selección.*

Las características principales que un ADL debería ofrecer para ser útil como vehículo de representación de la arquitectura de los sistemas son: permitir la representación de las entidades básicas, la descripción de arquitecturas dinámicas, la verificación de propiedades, permitir el desarrollo y evolución de la arquitectura.

En primer lugar el lenguaje debe permitir la representación de las entidades básicas. El término entidades básicas se refiere a los componentes, conectores, interfaces y puertos, los que combinados de forma adecuada pueden describir la arquitectura de un sistema de software.

El objetivo de hacer explícitos los aspectos arquitectónicos del software es facilitar el desarrollo y evolución de sistemas complejos. Debido al carácter inherentemente dinámico y reconfigurable de muchos de estos sistemas, la capacidad de describir los aspectos que rigen la evolución de una arquitectura es un requisito básico de cualquier ADL[MR97].

El éxito del diseño arquitectónico depende en gran medida de que se logre realizar especificaciones precisas de los sistemas, sus componentes y las interacciones entre ellos. Los requisitos referidos al análisis abordan la necesidad de servir de apoyo al razonamiento, tanto automatizado como no automatizado, acerca de las descripciones arquitectónicas. Estas descripciones modelan a menudo complejos sistemas de software concurrentes o distribuidos. Según Robert J. Allen "...la comprobación de las propiedades de los sistemas en las etapas iníciales del desarrollo, como es la de diseño arquitectónico, reducen substancialmente los costes de los errores"[AG97].

El argumento más utilizado para el desarrollo y uso de modelos y descripciones arquitectónicas del software es que son necesarios para cubrir la brecha existente entre los diagramas informales de cajas y líneas y la implementación final de dichos. El proceso de desarrollo de cualquier sistema de software pasa por una serie de refinamientos sucesivos, en los que el sistema se representa a diferentes niveles de abstracción que lo van acercando de manera progresiva desde la especificación a la implementación. Esto es lo que se conoce con el nombre de desarrollo iterativo e incremental, y es la base de las de modelos de ciclo de vida del software y del proceso de desarrollo. Nenad Medvidovic Richard N. Taylor plantean que "...un buen ADL debería proporcionar mecanismos para el refinamiento de la arquitectura y sus componentes que faciliten este proceso de desarrollo"[TMM00]. La arquitectura evoluciona para reflejar la evolución de un sistema de software concreto, y también para dar lugar a familias de sistemas relacionados. Esta posibilidad de hacer evolucionar el diseño arquitectónico es precisamente la que permite su reutilización en

diferentes contextos.

La tabla 2.4 refleja el cumplimiento de las características anteriormente mencionadas por los ADL analizados en el capítulo anterior, las cuales constituirán los criterios de selección del ADL.

Tabla 2.4: Lenguajes de descripción de la arquitectura de software más usados..

	Representación de Entidades	Dinamismo	Verificación de propiedades	Desarrollo y reutilización
UniCon	Componentes y Conectores	No	No	No
Wright	Componentes y Conectores	No	Parcialmente	No
Rapide	Componentes	Limitado	Parcialmente	Si
Jacal	Componentes y conectores	No	No	No
Darwin	Componentes	Si	Parcialmente	No
LEDA	Componentes	Si	Si	Si
Acme	Componentes y conectores	No	Si	Si
C2	Componente	Limitado	No	Si
Aesop	Componentes y conectores	No	No	Si

Para la ponderación de los criterios empleados, son comparados entre ellos para determinar su importancia relativa, se determina a través de una matriz de dimensión $n \times n$, comúnmente denominadas matriz de comparación por pares y representada como A. En esta matriz cada elemento A_{ij} representa la importancia relativa que tiene el criterio i sobre el criterio j. Este grado de importancia se mide con la escala mostrada en la tabla 2.5.

Teniendo en cuenta los criterios de selección para el ADL se elabora la matriz de comparación por pares y se determina la ponderación que toma cada criterio, la cual se calcula a partir del promedio de cada fila de la matriz de comparación luego de ser normalizada, estos elementos son mostrados en la tabla 2.6.

Posteriormente se realiza una comparación por pares de cada una de las alternativas para los criterios de comparación. Los resultados de dicha comparación se muestran en las tablas 2.7 ,2.8,2.9 y 2.10.

TABLA 2.5: Importancia relativa de los criterios en la matriz de comparación por pares.

Definición	Valor Relativo	Valor Recíproco
El mismo grado de preferencia o importancia	1	1
Un grado moderado de preferencia o importancia	3	0.33
Un fuerte grado de preferencia o importancia	5	0.20
Un grado muy fuerte de preferencia o importancia	7	0.14
Un grado extremadamente fuerte de preferencia o importancia	9	0.11
Valores intermedios entre puntos adyacentes de la escala	2, 4, 6, 8	0.5, 0.25, 0.16, 0.12

TABLA 2.6: Matriz de comparación por pares de los criterios de selección para el ADL.

	Representación de Entidades	Dinamismo	Verificación de propiedades	Desarrollo y reutilización	Ponderación
Representación de Entidades	1	7	5	3	0,55
Dinamismo	0,14	1	0,2	0,33	0,06
Verificación de propiedades	0,2	5	1	0,2	0,14
Desarrollo y reutilización	0,33	5	3	1	0,26

La ponderación de cada una de las matrices de comparación por pares de las alternativas forma una matriz de pesos globales, al ser multiplicada dicha matriz por la matriz de ponderación de los criterios se obtiene como resultado el nivel de selección de cada uno de los lenguajes. En este caso el lenguaje con un mayor nivel de satisfacción de los criterios resulta Acme, en la tabla 2.11 se muestran los resultados.

Tabla 2.7: Matriz de comparación por pares de las alternativas para el criterio representación de entidades.

	Uni-Con	Wright	Rapi-de	Ja-cal	Dar-win	LE-DA	Acme	C2	Ae-sop	Pon-deración
Uni-Con	1	1	5	1	5	3	1	5	1	0,16
Wright	1	1	5	1	5	3	1	5	1	0,16
Rapide	0,2	0,2	1	0,2	1	0,14	0,2	0,14	0,2	0,03
Jacal	1	1	5	1	5	3	1	5	1	0,16
Darwin	0,2	0,2	1	0,2	1	0,14	0,2	1	0,2	0,03
LEDA	0,33	0,33	7,14	0,33	7	1	0,2	7	0,2	0,10
Acme	1	1	5	1	5	5	1	5	1	0,17
C2	0,2	0,2	7,14	0,2	1	0,14	0,2	1	0,2	0,05
Aesop	1	1	5	1	5	5	1	5	1	0,17

2.2.2 Técnicas de priorización de requisitos

Pueden encontrarse un grupo de métodos que permiten priorizar requisitos, dentro de estos, Nacy Mead investigadora del SEI tras un estudio realizado [Mea06] afirma que "..los más utilizados son el árbol binario de búsqueda, técnica de asignación numérica, planeando el juego, el método de los 100 puntos, el método de Wiegers, framework de priorización de requisitos y el proceso de jerarquía analítica".

La priorización en el modelo se lleva a cabo en dos etapas, durante la tormenta de ideas con los clientes como punto de partida del modelo y durante la selección de los conductores arquitectónicos en cada una de las iteraciones.

El método de Wiegers, la técnica de asignación numérica y el proceso de jerarquía analítica son métodos complejos de utilizar cuando el número de candidatos tiende a elevarse y los involucrados en la selección tienden a ser indecisos [LK04] como es el caso en el ambiente donde se aplicará el modelo.

Para la primera etapa de priorización se aprovechará el marco de la tormenta de ideas y se utilizará el método de los 100 puntos el cual consiste en asignarle 100 puntos a cada uno de los involucrados en el proceso de priorización, estos a su vez lo distribuyen entre los requisitos asignándole la mayor puntuación a los que mayor prioridad tengan según su opinión. Una vez realizada la votación se haya el promedio de cada uno de los requisitos y ordenan de mayor a menor quedando ordenados de acuerdo a la prioridad.

TABLA 2.8: Matriz de comparación por pares de las alternativas para el criterio dinamismo.

	Uni-Con	Wright	Rapi-de	Ja-cal	Dar-win	LE-DA	Acme	C2	Ae-sop	Pon-deración
Uni-Con	1	1	0,2	1	0,11	0,11	1	0,2	1	0,03
Wright	1	1	0,2	1	0,11	0,11	1	0,2	1	0,03
Rapide	5	5	1	5	0,2	0,2	5	1	5	0,11
Jacal	1	1	0,2	1	0,11	0,11	1	0,2	1	0,03
Darwin	9,09	9,09	5	9,09	1	1	9	5	9	0,29
LEDA	9,09	9,09	5,00	9,09	7	1	9	5	9	0,36
Acme	1	1	0,2	1	0,11	0,11	1	0,2	1	0,03
C2	5	5	1,00	5	0,2	0,20	5	1	5	0,11
Aesop	1	1	0,2	1	0,11	0,11	1	0,2	1	0,03

La segunda etapa de la priorización será ejecutada en cada una de las iteraciones del modelo, por lo cual se necesita de un método fácil de implementar y en el que se llegue de forma rápida al resultado final, la técnica seleccionada es la de priorización basada en árboles binarios de búsqueda, la cual ofrece como principal ventaja que minimiza el número de comparaciones [Ahl05] ; está técnica se basa en los siguientes pasos:

1. Se tomará el primer requisito de la lista y se adicionará como nodo del árbol.
2. Se tomará el siguiente requisito se comparará con el nodo raíz, si este requisito tiene mayor prioridad que el nodo raíz, se procederá a comprar con el nodo derecho en caso contrario con el nodo izquierdo.
3. Si el nodo con el que se compara está vacío el requisito pasa a ocupar la posición de ese nodo.
4. Se repite el paso 2 y 3 mientras la lista no esté vacía.

Una vez ejecutada la técnica anterior se obtendrá un árbol binario con los requisitos, dicho árbol se recorre en pre orden para listar todos los requisitos en orden descendente de prioridad.

2.2.3 Método visual para la selección de conductores arquitectónicos

El método tiene como principal objetivo seleccionar en cada una de las iteraciones aquellos requisitos que pasarán a formar parte de los conductores arquitectónicos en esa etapa.

Tabla 2.9: Matriz de comparación por pares de las alternativas para la representación de propiedades.

	Uni-Con	Wright	Rapide	Jacal	Darwin	LEDA	Acme	C2	Aesop	Ponderación
Uni-Con	1	0,2	0,2	1	0,2	0,11	0,11	1	1	0,03
Wright	5	1	1	5	1	0,2	0,2	1	1	0,07
Rapide	5	1	1	5	1	0,2	0,2	5	5	0,10
Jacal	1	0,2	0	1	0,2	0,11	0,11	1	1	0,03
Darwin	5	1	1	5	1	0,2	0,2	5	5	0,10
LEDA	9	5	5	9	7	1	1	9	9	0,31
Acme	9	5	5	9	5	1	1	9	9	0,30
C2	1	1	0,2	1	0,2	0,11	0,11	1	1	0,03
Aesop	1	1	0,2	1	0,2	0,11	0,11	1	1	0,03

Toma como entrada los requisitos del sistema priorizados por parte del cliente y por parte del equipo de arquitecto.

Posteriormente se determina el índice de prioridad dado por los arquitectos (IP_{AR}) y por los clientes (IP_{CR}) para cada uno de los requisitos.

Tomando a L como la longitud de la lista y P_{Rj} como la posición del requisito Rj en la lista, se procede a calcular el índice de prioridad de cada requisito a través de la siguiente fórmula:

$$IP_{ARj} = (5/L_{RA}) * (L_{RA} - P_{Rj})$$

$$IP_{CRj} = (5/L_{RC}) * (L_{RC} - P_{Rj})$$

Siendo:

- L_{RA} el listado de los requisitos priorizados por el equipo de arquitectos.
- L_{RC} el listado de los requisitos priorizados por los clientes.

El siguiente paso consiste en determinar para la iteración en que se encuentre el grado de dificultad de satisfacerlo (G_{Rj}) tomando como escala la mostrada en la tabla 2.12.

TABLA 2.10: Matriz de comparación por pares de las alternativas para el desarrollo y reutilización.

	Uni-Con	Wright	Rapi-de	Ja-cal	Dar-win	LE-DA	Acme	C2	Ae-sop	Pon-deración
Uni-Con	1	1	0,11	1	1	0,11	0,11	0,11	0,11	0,02
Wright	1	1	0,11	1	1	0,11	0,11	0,11	0,11	0,02
Rapide	9	9	1	1	9	1	1	1	1	0,16
Jacal	1	1,0	1	1	1	0,11	0,11	0,11	0,11	0,04
Darwin	1	1	0	1	1	0,11	0,11	0,11	0,11	0,02
LEDA	9	9	1	9	7	1	1	1	1	0,18
Acme	9	9	1	9	9	1	1	1	1	0,19
C2	9	9	1	9	9	1	1	1	1	0,19
Aesop	9	9	1	9	9	1	1	1	1	0,19

TABLA 2.11: Nivel de selección de los ADL.

	Nivel se selección
Acme	**0,18**
LEDA	0,16
Aesop	0,15
Wright	0,10
Jacal	0,10
UniCon	0,10
C2	0,08
Rapide	0,08
Darwin	0,05

En este momento cada requisito relacionado cuenta con dos índices de prioridad, una por parte del cliente y otro por el grupo de arquitectos; además de su grado de dificultad para satisfacerlo. Auxiliándose de un gráfico burbuja, tomando como eje Y el índice de prioridad dado a los requisitos por el cliente (IP_{CRj}), en el X el índice asignado por los arquitectos (IP_{ARj}) y como diámetro de las burbujas el grado de dificultad de satisfacer los requisitos (G_{Rj}) se realiza la selección, esta está asociada a la sección del gráfico donde se encuentre y el diámetro de la burbuja tal y como se muestra en la figura 2.3, deben seleccionarse cinco o seis requisitos según estudios realizados por un grupo de investigadores del SEI [WBB+].

Tabla 2.12: Escala para determinar el grado de dificultad de satisfacer los requisitos.

Valor	Descripción
1 (Alto)	Asignarle a un requisito este grado de dificultad significa que satisfacerlo implicará altos esfuerzos teniendo en cuenta el estado actual de la arquitectura.
2 (Moderado)	Asignarle a un requisito este grado de dificultad significa que satisfacerlo implicará moderados esfuerzos teniendo en cuenta el estado actual de la arquitectura.
3 (Bajo)	Asignarle a un requisito este grado de dificultad significa que satisfacerlo implicará bajos esfuerzos teniendo en cuenta el estado actual de la arquitectura.

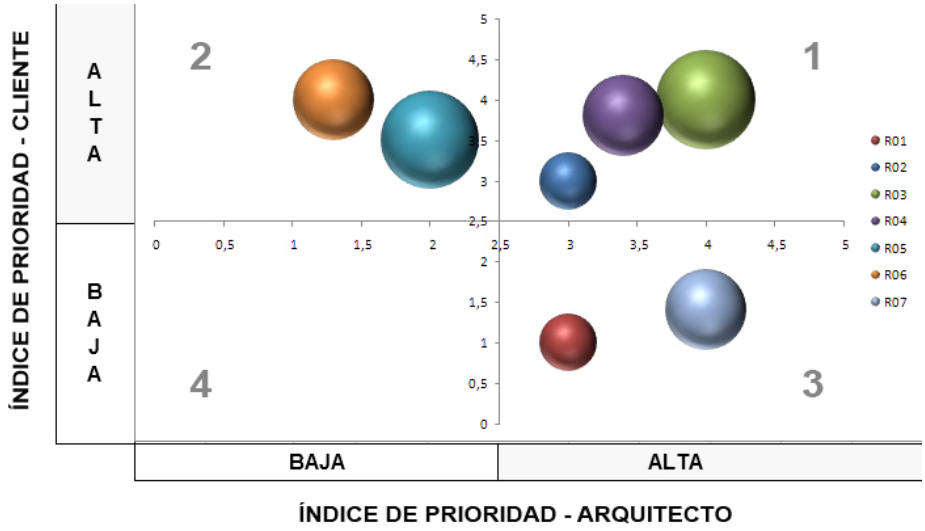


Figura 2.3: Gráfico para la selección para los conductores arquitectónicos.

2.3 Conclusiones parciales

- Fue descrito el modelo propuesto a través de las actividades presentes en el mismo, abordando para cada una de estas los elementos fundamentales que deben estar presentes y la forma de desarrollarlas.
- Mediante el uso de el proceso de jerarquía analítica (AHP) como técnica de decisión multicriterio fue seleccionado Acme el ADL indicado a ser utilizado en el modelo propuesto como mecanismo para la descripción de la arquitectura.
- Dar una correcta priorización a los requisitos e incluir en estos una visión tanto del arquitecto de software de como los clientes, aumenta desde etapas tempranas las probabilidades de arribar a las soluciones arquitectónicas deseadas.

3

Evaluación del modelo

En el presente capítulo se realiza una evaluación el modelo propuesto a ser utilizado en el proceso de definición de la arquitectura de los sistemas de gestión de información biomédica desarrollados en la facultad seis, dicha evaluación se ha dividido en tres partes fundamentales.

- Valoraciones generales, para analizar algunos aspectos relevantes del modelo, mostrando de esta forma sus fortalezas.
- Comparación con otros métodos basados en la arquitectura, reflejando el nivel de cumplimiento de ciertos indicadores en el modelo con respecto a dichos métodos.
- Aplicación del modelo, donde se describe los resultados que se obtuvieron tras aplicar el modelo en la definición de la arquitectura de dos proyectos de la facultad seis.

3.1 Valoración general

El modelo definido presenta algunos elementos que constituyen fortalezas, contribuyendo estos a un mayor grado de calidad en las arquitecturas definidas a través de dicho modelo. A continuación se realizará un análisis de estos elementos.

Como primer aspecto puede citarse que *están presentes los criterios del cliente en todo momento*, posibilitando un aumento de las probabilidades de que la arquitectura definida cumpla con todos los requisitos establecidos por este, siempre que no afecten la calidad de la solución.

¿De qué forma se logra?, en la primera actividad que se desarrolla (ver epígrafe 2.1.1) el cliente prioriza todos los requisitos del sistema y puede cerciorarse de que no falte niguno. Posteriormente, cuando se van a seleccionar los conceptos de diseños apropiados para descomponer el sistema se tiene en cuenta esta clasificación. Claro está que este no es el único indicador de prioridad de los requisitos que se tiene presente, sino que unido a la prioridad dada por el grupo de arquitectos y el grado de dificultad de satisfacerlo, costituyen las tres dimensiones aplicadas a la priorización de los requisitos (figura 3.1) en cada una de las iteraciones.

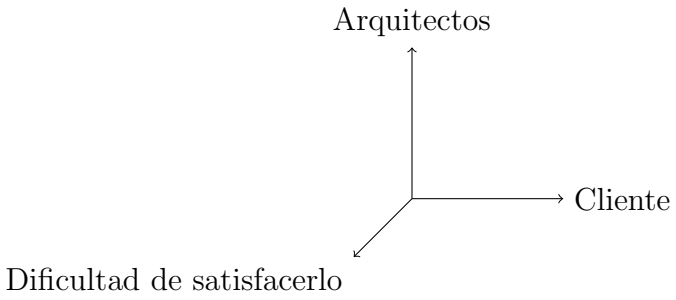


FIGURA 3.1: Dimensiones para la priorización de los requisitos

Otro aspecto a tener en cuenta es que *el proceso de definición de la arquitectura es iterativo incremental*. Como resultado de cada una de estas iteraciones se obtiene un incremento de la arquitectura definida mediante la descomposición de nuevos componentes aplicando estrategias arquitectónias apropiadas en cada caso.

En cada una de estas *iteraciones* se selecciona un componente del sistema a descomponer, constituyendo este el centro de la iteración; posibilitando manejar la complejidad del sistema, apuntando a la resolución de los problemas por partes. Otra de las ventajas ofrecida por ser un modelo *iterativo* es que permite ir ajustando y mejorando el diseño

definido.

En el modelo *están presentes las tres fases fundamentales del proceso de definición de la arquitectura* (diseño, evaluación y descripción)por lo que puede afirmarse que es un método completo.

En cuanto a la evaluación de la arquitectura esta puede clasificarse en dos tipos teniendo en cuenta la etapa en la que se realice, evalución temprana o evaluación tardía [CKK02]. La evaluación tardía consiste en realizar la evaluación de la arquitectura cuando ésta se encuentra establecida y la implementación se ha completado, este tipo de evaluación tiene como principal desventaja que ya la arquitectura está definida y si el proceso de evaluación arroja malos resultados trairía consigo un rediseño de la arquitectura.

Sin embargo, una evaluación *temprana* realizada desde las fases iniciales de diseño y a lo largo del desarrollo eleva las probabilidades de éxito, en el modelo se realiza este tipo de evaluación.

La mayoría de los métodos basados en la arquitectura conocidos definen el *qué* debe harcerse en cada una sus actividades y no el *cómo*, esto provoca que las personas con pocos conocimientos de arquitectura de software en ocasiones ejecutan el método de manera errónea. El modelo *incluye propuestas de técnicas a utilizar* las cuales fueron seleccionadas por su fácil comprensión y adecuación a la etapa en la que se debe emplear.

3.2 Comparación con otros métodos

Los métodos basados en la arquitectura por su objetivo pueden ser clasificados en dos grandes grupos, aquellos que permiten diseñar una arquitectura y los métodos colaterales, llamándosele de esta forma a los métodos que tienen como principal objetivo evaluar la arquitectura, describirla o identificar y priorizar tanto los requisitos como los escenarios de los atributos de calidad. Por tal razón la comparación que se realizará con el modelo propuesto en la presente investigación será dividida en dos partes fundamentales: comparación con los métodos para el diseño arquitectónico y comparación con los métodos de evaluación.

El ciclo de actividades presentes en el diseño de una arquitectura pueden ser descrita graficamente como se indica en la figura 3.2.

Hay tres actividades fundamentales [HKN+05], esá son:

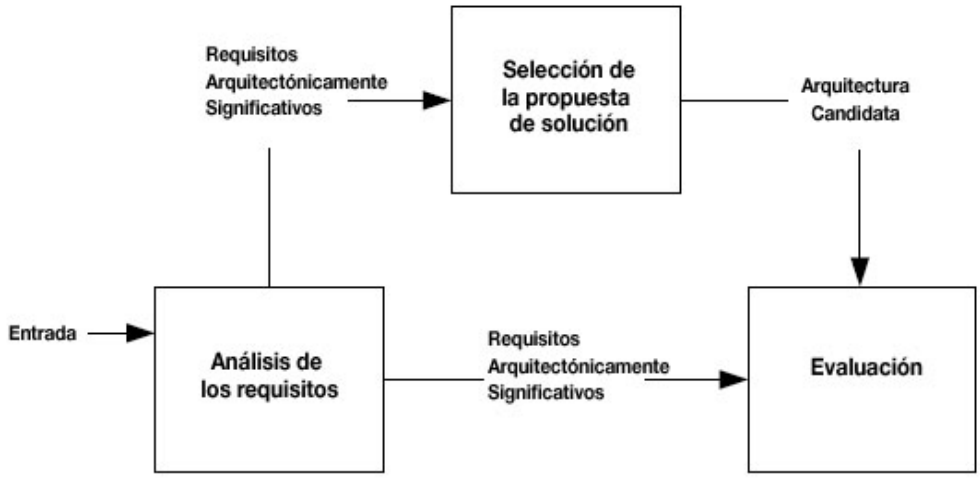


FIGURA 3.2: Actividades en el diseño de la arquitectura.

- Análisis de los requisitos, donde se seleccionan aquellos elementos que se tendrán en cuenta para diseñar la arquitectura.

- Selección de la propuesta de solución, actividad en la cual se definen los componentes que formarán parte de la arquitectura.

- Evaluación, dedicada a la comprobación del cumplimiento de los requisitos del sistema.

La entrada la constituyen aquellos elementos que se toman como base para la definición de la arquitectura, estos pueden ser requisitos, cualidades del sistema u objetivos de la organización para la cual se desarrollará.

Los requisitos arquitectónicamente significativos son aquellos que influyen sobre la arquitectura, estos no tienen por qué estar recogidos como requisitos del sistema, sin embargo constituyen elementos claves para el éxito del diseño por lo cual se debe ser muy cuidadoso en su selección.

La arquitectura candidata es aquella que se forma a partir de la propuesta de solución, se clasifica como candidata pues debe pasar por un proceso de evaluación antes de ser seleccionada como solución.

En la tabla 3.1 y 3.2 se realiza una comparación entre los métodos basados en la

arquitectura vistos en el capítulo 1 y el modelo tomando como parámetros para está comparación las actividades desarrolladas en los mismos.

Posteriormente en la tabla 3.3 se presenta una comparación entre los métodos de evaluación Software Architecture Analysis Method (SAAM), Architecture Trade-off Analysis Method (ATAM), Active Reviews for Intermediate Designs (ARID), Cost-Benefit Analysis Method (CBAM) y el modelo.

Tabla 3.1: Comparación entre los métodos de diseño arquitectónico: Actividades.

Actividades	ADD	S4V	RUP	Modelo Propuesto
Análisis de los requisitos	A partir de los requisitos del sistema y los atributos de calidad selecciona conductores arquitectónicos en cada una de las iteraciones.	Se realiza a partir del análisis de los factores que influyen sobre la arquitectura a desarrollar dando una elevada importancia a la modularidad y flexibilidad del sistema.	Selecciona los casos de uso arquitectónicamente significativos, a su vez los prioriza dando un orden de selección a los mismos en cada una de las iteraciones.	Se seleccionan los requisitos arquitectónicamente significativos. De este conjunto se son escogidos los conductores arquitectónicos en cada una de las iteraciones basados en el método visual para la selección de conductores arquitectónicos.
Propuesta de solución	Selecciona los conceptos de diseño que satisfacen los conductores arquitectónicos y descompone el sistema basados en estos conceptos.	Identifican posibles estrategias arquitectónicas a emplear y seleccionan la más indicada a aplicar a partir de la comparación de las mismas.	Selección de las estrategias arquitectónicas que dan solución al caso de uso del sistema seleccionado en esa iteración.	Se realiza un a través de análisis de los conceptos de diseño que satisfacen los conductores arquitectónicos seleccionados.
Evaluación	Realiza un análisis del cumplimiento de los requisitos en cada una de las iteraciones como forma de evaluar la solución propuesta.	La evaluación se realiza a través de equipos revisores los cuales analizan la solución a partir de listas de chequeo.	Propone construir prototipos funcionales del sistema a desarrollar para comprobar si fueron cumplidos los requisitos.	En cada iteración se evalúa el cumplimiento de los requisitos definidos para el sistema.

Tabla 3.2: Comparación entre los métodos de diseño arquitectónico: Artefactos.

Actividades	ADD	S4V	RUP	Modelo Propuesto
Entrada	Requisitos funcionales, atributos de calidad y restricciones de diseño.	Estándares de desarrollo, característica del producto y factores tecnológicos que podrían influir sobre la arquitectura.	Casos de uso del sistema.	Requisitos funcionales, atributos de calidad y restricciones de diseño.
Requisitos arquitectónicamente significativos	Aquellos requisitos funcionales y atributos de calidad que influyen sobre la arquitectura del sistema.	Funcionalidades del sistema que puedan influir sobre la arquitectura del mismo.	Casos de uso arquitectónicamente significativos.	Aquellos requisitos funcionales y atributos de calidad que influyen sobre la arquitectura del sistema.
Arquitectura candidata	Colección de vistas en las que se refleja la estructura del sistema definidas a partir de patrones y estrategias arquitectónicas.	Se obtiene a partir de las 4 vistas arquitectónicas, las cuales representan restricciones de diseño que satisfacen las funcionalidades del sistema.	Se construye en cada una de las iteraciones donde se enriquecen las 4 vistas de la arquitectura a partir de la solución del caso de uso del sistema seleccionado en esa iteración.	Colección de vistas en las que se refleja la estructura del sistema definidas a partir de patrones y estrategias arquitectónicas.

Tabla 3.3: Comparación entre métodos de evaluación.

	SAAM	ATAM	ARID	CBAM	Modelo propuesto
Atributos de Calidad Contemplados.	Modificabilidad. Seguridad. Confiabilidad. Desempeño.	Modificabilidad. Funcionalidad.	Conveniencia del diseño evaluado.	Funcionalidad. Modificabilidad.	Conveniencia del diseño evaluado.
Objetos analizados	Estilos arquitectónicos. Documentación. Flujo de datos. Vistas Arquitectónicas.	Documentación. Vistas Arquitectónicas.	Especificación de los componentes.	Estilos Arquitectónicos. Documentación.	Documentación. Vistas Arquitectónicas
Etapas del proyecto en las que se aplica.	Luego de que el diseño de la arquitectura ha sido establecido.	Luego de que la arquitectura cuenta con funcionalidad ubicada en módulos.	A lo largo del diseño de la arquitectura.	Luego de que el diseño de la arquitectura ha sido establecido.	A lo largo del diseño de la arquitectura.
Enfoques utilizados.	Utility Tree y tormenta de ideas para articular los requerimientos de calidad. Análisis arquitectónico que detecta puntos sensibles, puntos de balance y riesgos.	Tormenta de ideas para escenarios y articular los requerimientos de calidad. Análisis de los escenarios para verificar funcionalidad o estimar el costo de los cambios.	Revisiones de diseños, lluvia de ideas para obtener escenarios.	Teoría "W.ªnálisis de escenarios.	Tormenta de ideas y revisiones del diseño en cada una de las iteraciones.

3.3 Aplicando el modelo

El modelo fue aplicado en la definición de la arquitectura de dos proyectos en la facultad seis de la Universidad de las Ciencias Informáticas, estos fueron:

- Sistema informático de Genética Médica.
- Sistema para la representación gráfica de árboles genealógicos.

En ambos casos el proceso de aplicación del modelo se dividió en tres etapas fundamentales:

1. Capacitar a los arquitectos del proyecto con el objetivo de que logren familiarizarse con el modelo y adquirir los conocimientos necesarios para aplicarlo.
2. Aplicar el modelo en el proceso de definición de la arquitectura.
3. Analizar los resultados obtenidos.

3.3.1 Sistema informático de Genética Médica

El Centro Nacional de Genética Médica (CNGM), perteneciente al sector de la salud, creado en el año 2003 con el objetivo fundamental de llevar a cabo acciones asistenciales, docentes y de investigación en el campo de los problemas de salud de carácter genético, encaminados a elevar la calidad de vida y el bienestar del pueblo cubano; hoy constituye el centro coordinador de una red de 184 centros asistenciales dispersos por toda la isla los cuales conducen el"Programa nacional para el diagnóstico, manejo y prevención de enfermedades genéticas y defectos congénitos".

La principal fortaleza de los servicios de genética médica en Cuba radica en la existencia de la red nacional que, desde la atención primaria de salud, se integra con los diferentes niveles de atención del sistema nacional de salud y con las restantes especialidades médicas, para dar amplia cobertura a las demandas asistenciales en este campo.

Dicha red de profesionales de la salud cubana realiza estudios clínicos genéticos a la población, posibilitando el desarrollo de investigaciones científicas que permiten mejorar

los niveles de vida de la población cubana, disminuir el impacto de las enfermedades con implicación genética y contribuir al desarrollo científico técnico del país.

Como ejemplo se puede mencionar el estudio de más de 58 000 parejas de gemelos llevado a cavo en el año 2004, también podría citarse el estudio clínico genético de las personas con discapacidad del año 2007 donde fueron estudiados más de 360 000 personas con discapacidad.

A lo largo de los años, el volumen de información ha ido aumentando considerablemente al punto que la Red Nacional de Genética Médica se enfrenta a problemas dentro de los cuáles se encuentran los siguientes:

- Dispersión de la información.
- Imposibilidad de monitorear la información en un corto período de tiempo.
- Falta de actualización de los datos de los estudios realizados.

Con el objetivo de dar solución a la problemática anterior y partiendo de la prioridad que tiene la informatización del sector de la salud, teniendo en cuenta los diversos estudios que se realizan en la Red Nacional de Genética Médica y cumpliendo con las normas establecidas en el desarrollo de software para la salud, se desarrolla un sistema con arquitectura modular, flexible y plenamente integrado con otras soluciones del sector de la salud; el Sistema Informático de Genética Médica.

Para la definición de la arquitectura se utiliza el modelo propuesto en la presente investigación, a continuación se realizará un análisis de los resultados obtenidos.

Para tener una idea de la efectividad del modelo propuesto para la definición y evaluación de la arquitectura se comparará con una versión de ese mismo sistema que se realizó previamente y que hubo que desarrollar nuevamente. En la tabla 3.4 se presentan los criterios de comparación así como las características de estos en cada una de las versiones del sistema.

Con el objetivo de mostrar de forma gráfica la madurez que alcanzó la arquitectura del sistema alasMEDIGEN, en la figura 3.3 se muestra un gráfico radial con los mismos indicadores presentados en las tablas anteriores.

A la vez también se le realizaron un grupo de pruebas al sistema alasMEDIGEN que ratificaron la calidad de la solución desarrollada.

La fiabilidad, se comprobó su cumplimiento a través de la liberación de sistema por CaliSoft y la realización de pruebas de aceptación con el cliente, además de estar sometido a pruebas constantes por parte del equipo de trabajo durante su desarrollo.

La usabilidad, con el objetivo de lograr el cumplimiento de esta característica fue diseñado cumpliendo con los estándares propuestos para los sistemas informáticos del sector de la salud e incorporándole una interfaz atractiva e interactiva.

Un equipo del grupo de Calidad de Software realizó pruebas rendimiento con la herramienta jmeter, utilizando un número de usuario mayor al pronosticado obteniéndose resultados satisfactorios, en la tabla 3.5 se muestran los mismos.

La mantenibilidad se logró gracias al diseño modular con bajo acoplamiento y la utilización de buenas prácticas de arquitectura como el uso de patrones arquitectónicos y la separación de responsabilidades; esto también posibilitó que en el período actual se le agregaran dos módulos nuevos los que se encuentran en pruebas de aceptación.

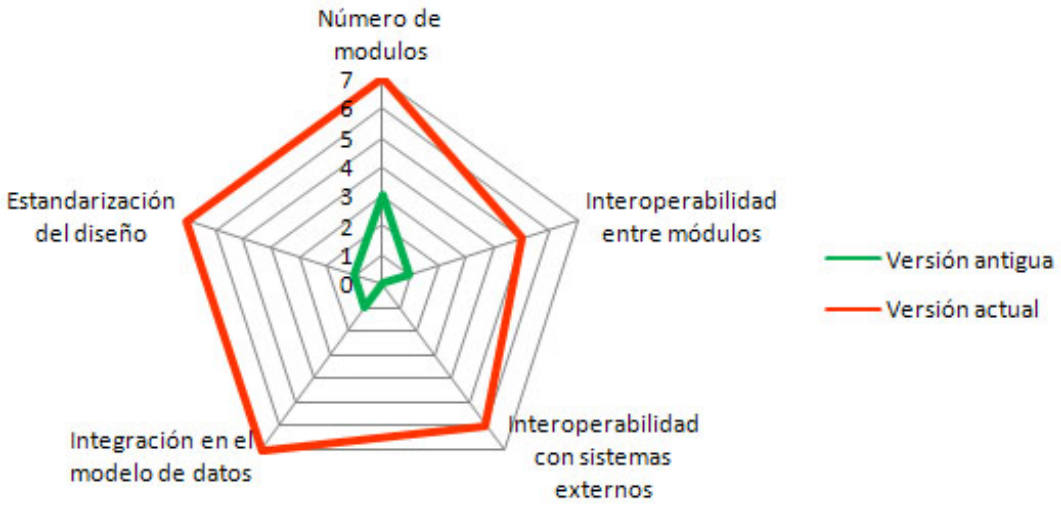


Figura 3.3: Comparación entre las dos versiones del sistema.

Tabla 3.4: Comparación entre los sistemas desarrollados.

	Número de módulos que contempla la solución	Interoperabilidad entre módulos	Interoperabilidad con sistemas externos	Modelo de datos	Estandarización del diseño	Tiempo de desarrollo
Versión desarrollada sin definir la arquitectura a través del modelo.	Gemelo, Discapacitado Intelectual, Discapacidad Física	No existe	No existe	Los módulos desarrollados contaban con bases de datos distintas a pesar existir información común que era gestionada en los tres módulos sin existir una integridad referencial entre estas.	Entre los módulos no existe una estandarización.	24 meses
Versión desarrollada definiendo la arquitectura a través del modelo.	Historia Clínica, Gemelo, Discapacitado Intelectual, Discapacidad Física, Tele consulta, Malformaciones, Enfermedades Genéticas.	Existe	Integrado a los registros del Sistema Informático de la Salud.	Todos los módulos comparten un modelo de datos común.	El desarrollo de los módulos fue estandarizado, desde la interfaz gráfica hasta en el uso de las tecnologías.	8 meses

Tabla 3.5: Resultado de las pruebas de rendimiento realizadas al sistema.

No. Uusarios	# Muestras	Media	Mediana	Línea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
100	207	12692	13875	23703	328	37672	0.0	1.6/sec	8716.8
200	205	20498	15844	54000	343	61375	1.89	1.4/sec	6409.3
300	417	23551	18312	45625	343	70984	0.0	1.0/sec	5122.0

3.3.2 Sistema para la representación gráfica de árboles genealógicos

Los árboles genealógicos familiares, son el recurso más utilizado por los profesionales de la genética médica desde hace un siglo, se realizan a un paciente referido a la consulta ante un posible diagnóstico genético, para evaluar la transmisión del rasgo o enfermedad en la familia y asesorar sobre el riesgo a los miembros que lo soliciten. La identificación de familias con varios miembros afectados por estas enfermedades crónicas, permite realizar una mejor caracterización clínica de la misma, identificar tempranamente a otros miembros de la familia enfermos y abordar con un propósito preventivo a los demás familiares en riesgo [Mar08].

El Sistema para la representación gráfica de árboles genealógicos (alasARBOGEN) tiene como principal objetivo convertirse en la herramienta utilizada por los doctores especialistas en genética clínica de la red nacional de genética médica; a través de la cual puedan representar el árbol genealógico asociado a cada paciente, que no es más que la representación de la familia mediante dibujos y diagramas que ayudan a un mejor análisis e interpretación de las afectaciones o enfermedades genéticas de un individuo y su trascendencia en la familia.

alasARBOGEN posibilitará elevar el nivel de las investigaciones realizadas por los especialistas en genética clínica, haciendo uso de la red e integrado al Registro Informatizado de Salud Cubano.

Para la definición de la arquitectura del alasARBOGEN se utilizó el modelo propuesto en la presente investigación. En estos momentos el sistema se encuentra en fase de implementación para posteriormente relizarle las pruebas reglamentadas por el grupo de calidad de la Universidad.

Finalmente la estructura definida para el sistema es la que se muestra en la figura 3.4 la cual puede ser descrita como:

- Dos sistemas diseñados bajo el estilo arquitectónico Modelo Vista Controlador los cuales son desarrolladas utilizando el framework Spring.
  - Una herramienta desktop que será utilizada por los especialistas en genética clínica, la cual puede trabajar aislada a la red y sincronizar toda su información en etapas posteriores.
  - Una herramienta web del lado del servidor.

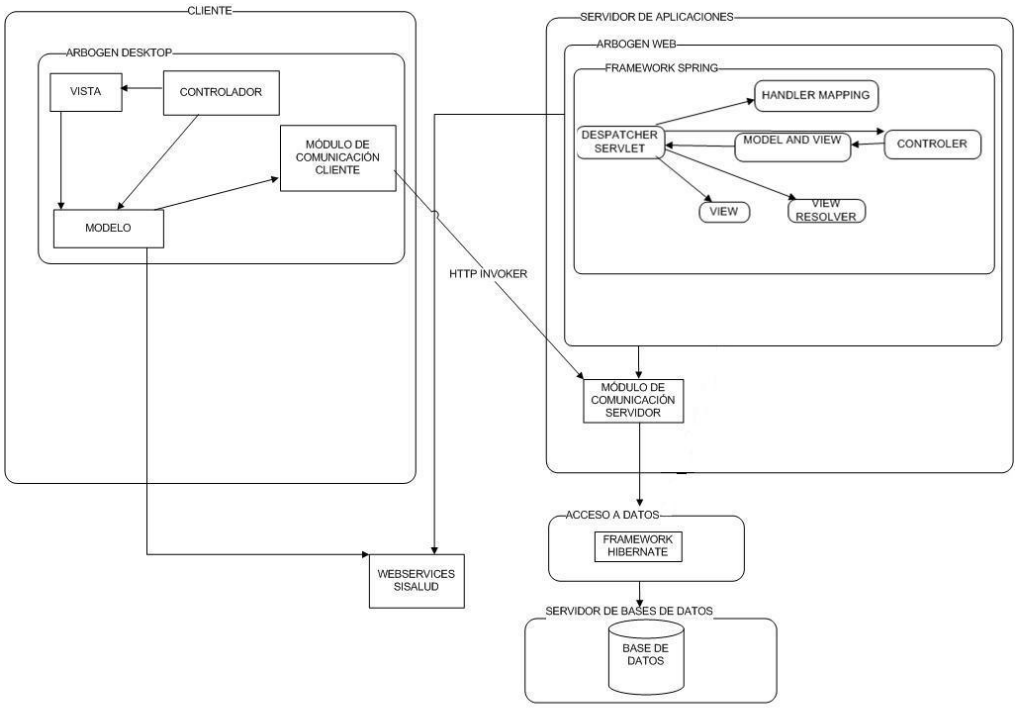


Figura 3.4: Comparación entre las dos versiones del sistema.

- Dos componentes de comunicación que posibilitan la sincronización por protocolos seguros de la información entre el cliente y el servidor.

- Una capa de acceso a datos implementado utilizando el framework Hibernate.

Al estar el sistema en fase de implementación no se pudo probar al nivel del sistema descrito en el epígrafe 3.3.1, en su lugar se realizó una evaluación de la arquitectura definida utilizando el método ATAM, el cual es considerado un método robusto para evaluar arquitecturas de software [SRR08]. Una vez realizada la evaluación se obtuvo como resultado que la arquitectura definida cumple con todos los requisitos establecidos para el mismo.

Otro aspecto ha destacar es que el equipo de arquitectos que participó en la definición de la arquitectura de este sistema no contaba con experiencias previas en este proceso. Apoyados en el modelo propuesto en esta investigación, profundizando los conocimientos adquiridos en la asignatura de Ingeniería de Software y auxiliándose en ocasiones del

criterio de expertos lograron una solución arquitectónica robusta que cumple con todos las espectativas trazadas.

3.4 Conclusiones parciales

- El modelo propuesto reduce el nivel de conocimiento requerido para definir al arquitectura de los sistemas, además permite representar la arquitectura del sistema de forma clara a través del ADL y las vistas definidas.
- La aplicación práctica del modelo ha permitido definir la arquitectura en dos sistemas del dominio de aplicación que cumplen con los atributos de calidad creando disciplina de trabajo y fomentando las habilidades del equipo de arquitectos.

Conclusiones

- El proceso de definición de la arquitectura de los sistemas de gestión de información biomédica desarrollados en la facultad seis no se realiza siguiendo buenas prácticas arquitectónicas, aspecto que ha incidido en el desarrollado un grupo de soluciones con un bajo cumplimiento de los atributos de calidad.
- En base a las características de los sistemas de gestión de información biomédica desarrollados y el análisis de algunos métodos basados en la arquitectura se define un modelo a ser utilizado en el proceso de diseño y evaluación de las arquitecturas.
- El modelo propuesto reduce el nivel de conocimiento requerido para definir la arquitectura de los sistemas, posibilita la definición de arquitecturas que cumplen con los atributos de calidad, crea disciplina de trabajo y fomenta las habilidades del equipo de arquitectos.

Recomendasiones

- Aplicar el modelo propuesto en la definición de la arquitectura de otros sistemas dentro y fuera de la facultad extendiendo el ámbito de la utilización del mismo.
- Desarrollar una herramienta para la aplicación de las técnicas de priorización utilizadas en las distintas fases del modelo, facilitando la ejecución de las actividades presentes en el mismo.
- Utilizar las vistas de la arquitectura para la generación de código fuente a partir de la arquitectura dirigida por modelos (MDA pos sus siglas en inglés).

Referencias Bibliográficas

[AG97] R.J. Allen and D. Garlan. A formal approach to software architecture, 1997.

[Ahl05] V. Ahl. An experimental comparison of five prioritization methods. *Master's Thesis, School of Engineering, Blekinge Institute of Technology, Ronneby, Sweden*, 2005.

[BCKK08] Len Bass, Paul Clements, Rick Kazman, and Mark Klein. Evaluating the software architecture competence of organizations. In *WICSA '08: Proceedings of the Seventh Working IEEE/IFIP Conference on Software Architecture (WICSA 2008)*, pages 249–252, Washington, DC, USA, 2008. IEEE Computer Society.

[BEL+03] M.R. Barbacci, R. Ellison, A.J. Lattanze, J.A. Stafford, C.B. Weinstock, and W.G. Wood. Quality Attribute Workshops (QAWs), 2003.

[bes00] Ieee recommended practice for architectural description of software-intensive systems. Technical report, 2000.

[BKB00] L. Bass, M. Klein, and F. Bachmann. Quality attribute design primitives. *Software Engineering Institute Technical Report CMU/SEI-2000-TN-017*, 2000.

[BZJ04] MA Babar, L. Zhu, and R. Jeffery. A framework for classifying and comparing software architecture evaluation methods. In *Software Engineering Conference, 2004. Proceedings. 2004 Australian*, pages 309–318, 2004.

[Can01] Jose Carlos Canal. *Un Lenguaje para la especificación y validación de arquitecturas de software*. PhD thesis, June 2001.

[Cer09] H. Cervantes. Mejorando las competencias arquitectónicas en una empresa mexicana de desarrollo de software. Technical report, Workshop, Arquitectura de Software., 2009.

[CGB+02] P. Clements, D. Garlan, L. Bass, J. Stafford, R. Nord, J. Ivers, and R. Little. *Documenting software architectures: views and beyond*. Pearson Education, 2002.

[CK03] P. Clements and R. Kazman. *Software Architecture in Practices*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 2003.

[CKK02] P. Clements, R. Kazman, and M. Klein. *Evaluating software architectures: methods and case studies*. Addison-Wesley Reading, MA, 2002.

[CKK+07] P. Clements, R. Kazman, M. Klein, D. Devesh, S. Reddy, and P. Verma. The duties, skills, and knowledge of software architects. In *Software Architecture, 2007. WICSA'07. The Working IEEE/IFIP Conference on*, pages 20–20, 2007.

[Cle00] P. Clements. Active Reviews for Intermediate Designs (CMU/SEI-2000-TN-009), Software Engineering Institute, 2000.

[FCK07] D. Falessi, G. Cantone, and P. Kruchten. Do Architecture Design Methods Meet Architects' Needs? In *Software Architecture, 2007. WICSA'07. The Working IEEE/IFIP Conference on*, pages 5–5, 2007.

[GMW00] D. Garlan, R.T. Monroe, and D. Wile. Acme: Architectural description of component-based systems. *Foundations of component-based systems*, pages 47–68, 2000.

[GPM05] A. Grimán, M. Pérez, and L. Mendoza. Estudio de la influencia de mecanismos arquitectónicos en la calidad del software. 2005.

[HKN+05] C. Hofmeister, P. Kruchten, RL Nord, H. Obbink, A. Ran, and P. America. Generalizing a model of software architecture design from five industrial approaches. In *Software Architecture, 2005. WICSA 2005. 5th Working IEEE/IFIP Conference on*, pages 77–88, 2005.

[KBWA94] R. Kazman, L. Bass, M. Webb, and G. Abowd. SAAM: A method for analyzing the properties of software architectures. In *Proceedings of the 16th international conference on Software engineering*, pages 81–90. IEEE Computer Society Press, 1994.

[KKC00] R. Kazman, M. Klein, and P. Clements. ATAM: Method for architecture evaluation. *CMU/SEI*, 2000.

[Kra95] P. Krachten. Mommy, where do software architectures come from? *1st International Workshop on Architectures for Software Systems, Seattle, WA*, 1995.

[LD05] J.E. López and J.J. Dolado. Estudio de los métodos de estimación: AHP y redes Bayesianas. 2005.

[LK04] L. Lehtola and M. Kauppinen. Empirical evaluation of two requirements prioritization methods in product development projects. *Software Process Improvement*, pages 161–170, 2004.

[Mar08] B. Marcheco. Genética médica y enfermedades crónicas: el camino de la prevención. *Revista cubana de genética comunitaria*, 2(2):3–4, 2008.

[Mea06] Nancy R. Mead. Requirements prioritization introduction. Technical report, Carnegie Mellon University, 09 2006.

[MR97] N. Medvidovic and D.S. Rosenblum. Domains of Concern in Software Architectures. In *Proceedings of the 1997 USENIX Conference on Domain-Specific Languages*, volume 246, 1997.

[Off02] J. Offutt. Quality attributes of Web software applications. *IEEE Software,*, pages 25–32, 2002.

[Q+02] A. Quiroga et al. Gestión de información, gestión del conocimiento y gestión de la calidad en las organizaciones. *Acimed*, 10(5):05, 2002.

[Rey04] Carlos Billy Reynoso. Introducción a la arquitectura de software. Conferencia, Universidad de Buenos Aires, Marzo 2004.

[RR06] S. Robertson and J. Robertson. *Mastering the requirements process*. Addison-Wesley Professional, 2006.

[RW05] Nick Rozanski and Eoin Woods. *Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives*. Addison-Wesley Professional, April 2005.

[Sha96] Mary Shaw. Procedure calls are the assembly language of software interconnection: Connectors deserve first-class status. In *ICSE '93: Selected papers from the Workshop on Studies of Software Design*, pages 17–32, London, UK, 1996. Springer-Verlag.

[SRR08] N. Sankar, B. Rajalashmi, and P. Rodriguez. Impact on quality attributes for evaluating software architecture using atam and design patterns. *Medwell Journals*, 7(3):126–129, 2008.

[TMM00] R.M. Taylor, N. Medvidovic, and N. Medvidovic. A classification and comparison framework for software architecture description languages. *IEEE Transactions on Software Engineering*, 26(1), 2000.

[TT05] T. Tsumaki and T. Tamai. A Framework for Matching Requirements Engineering Techniques to Project Characteristics and Situation Changes. *Proceedings of Situational Requirements Engineering Processes (SREP), Paris, France*, 2005.

[VL07] M Velázquez Leyva. Modelo para el desarrollo de software seguro en la bioinformática médica. Master's thesis, Universidad de las Ciencias Informáticas, 2007.

[VS08] A.V.F. Vázquez and G.H.A. Salinas. Un Enfoque para la Aplicación de las Tácticas de la Arquitectura con el Fin de Ampliar la Documentación de sus Calidades. 2008.

[WBB+] R. Wojcik, F. Bachmann, L. Bass, P. Clements, P. Merson, R. Nord, and B. Wood. Attribute-driven design (add). Technical report, Technical Report CMU/SEI-2006-TR-023, CMU SEI Pittsburgh, 2006.