UNIVERSIDAD CENTROCCIDENTAL

"LISANDRO ALVARADO"

ARQUITECTURA DE SOFTWARE PARA AUTOMATIZAR LOS REGISTROS ACADEMICOS EN LA UNIVERSIDAD CENTROCCIDENTAL "LISANDRO ALVARADO"

RAMON ANTONIO VALERA ARANGUREN

Barquisimeto, 2010

UNIVERSIDAD CENTROCCIDENTAL "LISANDRO ALVARADO"

DECANATO DE CIENCIAS Y TECNOLOGIA

DEPARTAMENTO DE SISTEMAS

ARQUITECTURA DE SOFTWARE PARA AUTOMATIZAR LOS REGISTROS ACADEMICOS EN LA UNIVERSIDAD CENTROCCIDENTAL "LISANDRO ALVARADO"

Trabajo Presentado Como Requisito Para Optar a la
Categoría de Agregado en el Escalafón del Personal Docente
y de Investigación

RAMON ANTONIO VALERA ARANGUREN

Barquisimeto, 2010

INDICE GENERAL

UNIVERSIDAD CENTROCCIDENTAL "LISANDRO ALVARADO"

DECANATO DE CIENCIAS Y TECNOLOGIA

DEPARTAMENTO DE SISTEMAS

ARQUITECTURA DE SOFTWARE PARA AUTOMATIZAR LOS REGISTROS ACADEMICOS EN LA UNIVERSIDAD CENTROCCIDENTAL "LISANDRO ALVARADO"

Autor(a): Ramón Antonio Valera Aranguren

RESUMEN

El uso de técnicas de Ingeniería de software en la construcción de proyectos de Software de amplio impacto, garantiza la calidad del mismo y su fácil adaptación para la incorporación nuevas funcionalidades y modificaciones menores, en este contexto la Arquitectura de Software es preponderante en las primeras etapas de desarrollo del sistema, pues ofrece una visión general y abstracta del sistema a desarrollar, una descripción a muy alto nivel de su estructura y control global, los protocolos de comunicación y sincronización utilizados, la distribución física del sistema y sus componentes, la arquitectura puede ser enriquecida por un conjunto de patrones que supervisen la composición de sus componentes y las restricciones de comunicación entre ellos. Estas técnicas de construcción de software pueden ser de gran valor cuando se aplican a procesos dinámicos y cambiantes como por ejemplo los relacionados a la gestión académica de una Universidad, en este caso, el presente trabajo se centra a generar una arquitectura de software que permita automatizar y estandarizar los procesos en las Unidades de Control de Estudio de de cada uno de los Decanatos que conforman la UCLA de manera incremental e iterativa, soportado en documentos agiles y usando patrones, como un paso previo a resolver los

problemas de interoperabilidad y coordinación que existen entre las diversas dependencias de la Universidad.

Palabras Claves: Arquitectura de Software, Lenguaje de Patrones, Patrón de Software.

INTRODUCCION

Los modelos de desarrollo de software organizan el conjunto de actividades necesarias para obtener productos de software de calidad, se centran en definir un proceso de desarrollo racional y controlable, pero de ninguna manera existe un modelo de desarrollo universal y único, existen múltiples variantes, y una u otra son más o menos efectivas en el contexto donde se empleen. Los modelos de desarrollo representan una guía de cómo y en qué orden deben realizarse cada una de estas actividades, y establecen el marco del ciclo de vida del software.

La propuesta arquitectural de software es un resultado esperado en cualquier modelo de desarrollo moderno, destinado a dar una visión general del sistema en etapas tempranas a diseñadores y desarrolladores, para tomar decisiones firmes y minimizadas en riesgo, en cuanto a la determinación de responsabilidades, división de esfuerzo dentro del equipo de trabajo, organización de la aplicación, puntos de coordinación, despliegue de aplicación, entre otros. XP es una metodología ágil muy empleada y efectiva, puesto que propone un equipo de trabajo horizontal dinamizado por la comunicación y la interacción continua entre desarrolladores y usuarios finales de la aplicación. Establece un conjunto de actividades que de manera incremental e iterativa permite construir productos de software, entre ellos la arquitectura de software.

Para el presente trabajo se recurre primordialmente a un modelo de desarrollo ágil, para que de manera sistematizada y rápida se pueda atacar una situación de problema en el área de gestión académica en la Universidad Centroccidental “Lisandro Alvarado” (UCLA).

La UCLA está constituida por Decanatos, los cuales son órganos administrativos y académicos a través de los cuales, la Universidad realiza sus funciones de docencia, investigación y extensión. Para que en los Decanato se puedan llevar a cabo estas actividades, se requiere de la administración de recursos tanto físicos (aulas, laboratorios, equipos, etc.), humanos (profesores, auxiliares docente, personal administrativo, obreros) como de tiempo. Así mismo dentro de la UCLA existe la Secretaría General, que es la Unidad Administrativa responsable de:

- Recibir, procesar y emitir información relacionada a la gestión universitaria, tanto dentro como fuera de la UCLA.

Para lograr este objetivo se apoya en una estructura organizativa, en la que convergen diversas direcciones y unidades, estas son:

- Dirección Docente.
- Dirección de Admisión y Control de Estudios.
- Archivo General.
- Unidad de Patrimonio.

Es indispensable para la secretaria general lograr que la información que maneja fluya de la manera más rápida posible hacia las distintas unidades y direcciones, ya que este es un insumo indispensable para la toma de decisiones en la estructura gerencial de la UCLA.

La Dirección de Admisión y Control de Estudios (DACE) es la encargada de planificar, coordinar y controlar todos los procesos relativos a admisión, registro, información y graduaciones del estudiantado, así como también, es responsable de velar por el cumplimiento de reglamentos, normas y disposiciones relacionadas a los programas de estudio que se desarrollan en los distintos decanatos de la UCLA.

La DACE lleva a cabo sus actividades con el apoyo de cada unidad de Registro Académico presente en cada Decanato de la UCLA. La DACE se encarga de

gestionar el ingreso y egreso de estudiantes a cada programa que imparte la universidad y la unidad Registro Académico se encarga, según lo especifica la resolución Nº 051-2004 del Consejo Universitario, de efectuar el seguimiento académico de toda la población estudiantil en el Decanato donde tiene jurisdicción, así mismo, es la responsable del control y custodia de expedientes de estudiantes, así como los trámites referidos a: inscripciones, retiros, cancelaciones, inclusiones y constancias.

La fuerte dependencia entre la DACE y cada unidad de registro académico hace necesario un flujo constante de información, para ello en la actualidad, la información se encuentra en aplicaciones locales en cada Decanato y son integrados por distintas vías en una aplicación del DACE, este proceso de integración se realiza en lotes, por lo que el repositorio de datos de la DACE nunca refleja la situación actual de los procesos académicos que se llevan a cabo los decanatos.

En fin existen un conjunto de problemas relacionados a la coordinación de los diversos procesos de gestión académica que llevan en conjunto la DACE y los diversos Registros Académicos de la UCLA, este problema tiene que ver con aspectos tecnológicos, integración de datos y procesos, seguridad, auditoria y elementos organizacionales. En este contexto el siguiente trabajo expone una propuesta arquitectural de software como solución a toda esta problemática, tomando como base un conjunto de necesidades descubiertas en un proceso de levantamiento de información sustentado en el manifiesto ágil, y empleando estrategias y técnicas de Ingeniería de Software.

METODOLOGIA DE TRABAJO

Como lo refiere Beck(2000), eXtreme Programming (XP) es una metodología ágil para equipos de desarrollo de software pequeños y medianos, diseñada para responder rápidamente a los cambios y lograr entregas de software, continuas, rápidas y de valor. Esta metodología es ideal para direccionar los esfuerzos necesarios en la intención de generar una arquitectura de solución para el problema de Registro Académico y Control de Estudio en UCLA, pues con pasos firmes y concisos se pueden lograr entregas de valor para avanzar en la consecución de los objetivos planteados, XP establece un ciclo de vida para el proyecto de software de seis fases: Exploración, Planificación de la Entrega de una Versión, Iteraciones, Producción, Mantenimiento y Muerte del Proyecto. (Ver Figura 1)

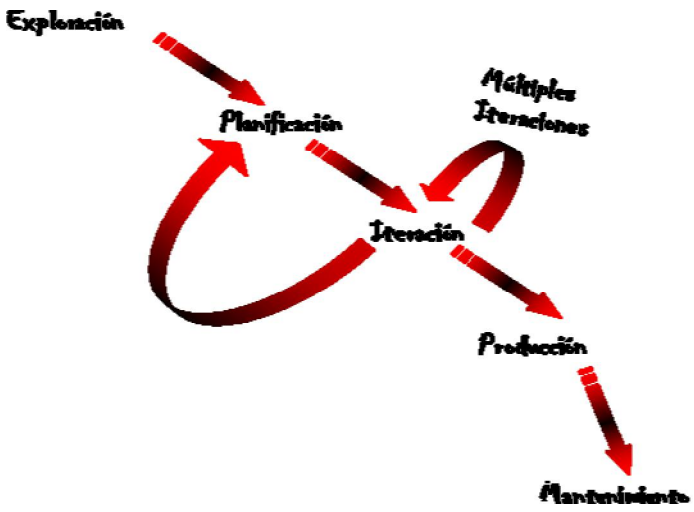

Figura 1. El proceso de desarrollo XP
Fuente: Beck (2000).

Exploración

Esta fase tiene como objetivo realizar un análisis de las necesidades en el cliente. Para el caso particular de este trabajo se recurre a la técnica JAD, que mediante un trabajo grupal entre especialistas de dominio, diseñadores y clientes finales, permite capturar las necesidades de los usuarios y en consecuencia enumerar las historias de usuario. Cada una de estas debe ser analizada para de tal manera que se establezca un estimado del esfuerzo requerido para convertirla en un componente de software.

Planificación

La fase de planificación incluye organizar el conjunto de funcionalidades recogidas en la fase de exploración en la forma de un plan de trabajo que muestra la prioridad y el tiempo estimado de culminación de cada funcionalidad, esto se hace en conjunto con el cliente, esto sirve de insumo para que el gerente de proyecto asignado por la empresa y el cliente, entienda el avance del proyecto. El plan de proyecto muestra el conjunto de iteraciones necesarias para construir el software y contiene los puntos de liberación de las versiones del software producido, para que el cliente final vaya comprobando y validando los avances obtenidos.

Iteración

Cada iteración representa un proceso de construcción en la que cada miembro del equipo desarrollador elabora los artefactos de software asignados. Dentro de las iteraciones de desarrollo se producen además de los componentes de software la siguiente documentación:

- Arquitectura de la Aplicación.
- Lenguaje de Patrones (Buenas Prácticas que regulan el comportamiento de los componentes de la arquitectura).
- Modelo de Dominio. (Conceptos Involucrados en el Desarrollo)

De acuerdo al plan de proyecto, se puede liberar versiones del código ejecutable del sistema para que sea probado y validado por el usuario final.

Mantenimiento

La fase de mantenimiento incluye las actividades de soporte para que bajo los parámetros acordados en el contrato de servicio se le de continuidad a los artefactos de software validados y verificados por el cliente final.

Fin del Proyecto

Se llega a esta etapa cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura.

DESCRIPCIÓN DEL PROBLEMA

Para conocer en detalle la problemática actual relacionada al funcionamiento de las Unidades de Registro académico y su interacción con DACE, se realizaron diversas actividades de reconocimiento y levantamiento de información soportado por la técnica Joint Application Development (JAD), Yatco (1999) indica que tiene como objetivo central facilitar la cooperación entre usuarios y analistas durante el desarrollo de sistemas. Los analistas de sistemas y los representantes funcionales realizan reuniones de trabajo con los usuarios directos para discutir las características de los sistemas objeto de estudio y, sobre la marcha de las mismas discusiones, se van trazando los modelos que permitirán definir los requerimientos funcionales de esos sistemas.

Las sesiones de JAD son de dos tipos: de adiestramiento y de trabajo. A su vez, las sesiones de trabajo se cumplen, normalmente como lo muestra la Figura 2, en tres etapas: revisión, formalización y validación.

En cada sesión de trabajo, a medida que van discutiéndose diferentes aspectos del sistema objeto de estudio, se van elaborando modelos de procesos y datos en borrador, haciendo uso de un pizarrón o de rotafolios, con el fin de que los participantes puedan confirmar, al equipo de desarrollo, si los modelos representan razonablemente bien los puntos por ellos expuestos

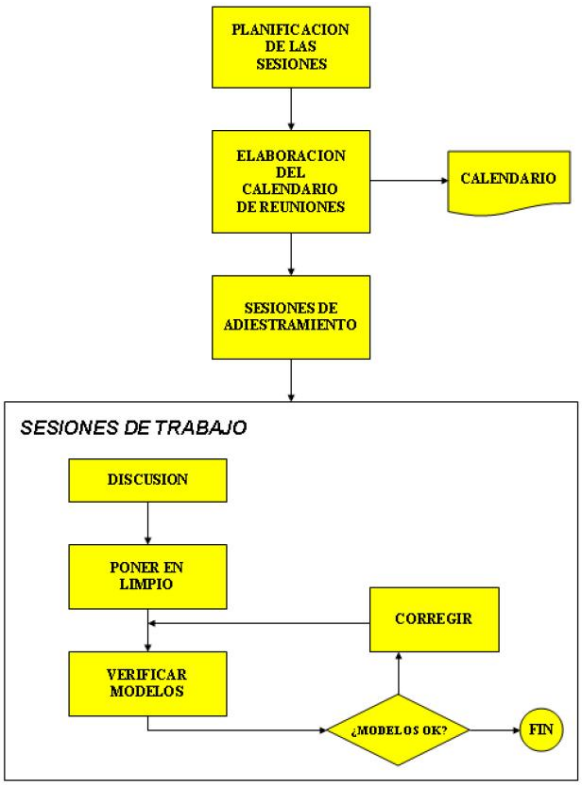

Figura 2. Etapas de la Técnica Joint Application Development (JAD)
Fuente: Beck (2000).

Las sesiones JAD realizadas con el personal de diversos Registros Académicos en la UCLA arrojaron las siguientes conclusiones:

- Las aplicaciones de los registros académicos están desarrolladas en ambientes tecnológicos diversos y no compatibles, es común ver trabajando en algunos sitios, equipos y aplicaciones de tecnología obsoleta, esto hace difícil los procesos de control, auditoria e integración, así mismo, no es posible garantizar la integridad y coherencia de los datos depositados en ellos.
- La mayoría de las aplicaciones instaladas en los Registros Académicos de la UCLA, fueron desarrolladas por el mismo personal de la unidad, esto se hizo sin ningún tipo de documentación, con lo cual su funcionamiento y operatividad esta directamente ligada al personal que lo desarrollo, esto es

inconveniente desde el punto de vista operativo, pues la ausencia del personal ocasionaría retrasos en la ejecución de los procesos.

- Existen problemas en los Registros Académicos de la UCLA que marcan el desempeño de los procesos académicos de los Decanato y que están muy relacionados al uso e implementación de tecnologías de información, entre ellos podemos citar:
    - Información no oportuna a las unidades responsables externas a las Oficina de Registro Académico de cada Decanato, el acceso a los sistema de información de cada Registro Académico está limitado solo al personal de la unidad, si la información estuviera al alcance de los Directores de Programa, Jefes de Departamento y Profesores, la toma de decisiones en los procesos académicos se realizaría de una forma más expedita.
    - Falta de Integridad y consistencia de los datos, este inconveniente está relacionado al uso de una infraestructura de base de datos obsoleta o al modelaje incorrecto de los datos en la Base de Datos.
    - No hay procesos estandarizados ni definidos, la ausencia de documentación acerca del desarrollo de los procesos académicos es evidente, es común ver distintas interpretaciones del reglamento general de evaluación en los distintos decanatos de la UCLA, así mismo se presentan procesos cuyo desarrollo se realiza sin algún procedimiento estandarizado.
    - Discrecionalidad del personal técnico en el manejo de los procesos de la unidad, la ausencia de políticas que regulen el uso y desarrollo de las tecnologías de información y comunicación en la UCLA, así como la ausencia en la estandarización de los procesos académicos, facilita al personal técnico que desarrolla las aplicaciones en la unidades de Registro Académico, discrecionalidad en el manejo e implementación de los procesos en sus sistemas, son evidentes las diferencias en las

interpretaciones acerca del desarrollo de los procesos en los distintos decanatos.

- Ausencia de mecanismos de auditoría, supervisión y control, en el desarrollo de los procesos académicos, de los cuales es responsable la unidad de Registro Académico.
- Ausencia de políticas de seguridad y respaldo para los datos de la unidad de Registro Académico.
- Dada la dependencia que tiene la DACE de la información que maneja las unidades de Registro Académico, la ausencia de políticas en aspectos claves como la seguridad, auditoria, control y respaldos, así como la discrecionalidad del personal técnico para el desarrollo de los procesos, afecta la confiabilidad y fiabilidad de los datos que le son enviados.

Se comprobó igualmente que los datos producidos en los procesos de gestión académica ejecutados en los Registros Académicos de los Decanatos y en la DACE, son consumidos o requeridos por otras dependencias estas son:

- Planificación Universitaria.
- Desarrollo Estudiantil.
- Dirección de Deportes.
- Dirección de Cultura.
- Dirección de Biblioteca.
- Vicerectorado Administrativo.

BASES TEORICAS

A continuación se presentan un conjunto de conceptos y definiciones relativos al Área de Ingeniería de Software, con la idea de construir un marco conceptual que apoye el desarrollo del presente trabajo, en este sentido se comenzará por entender el uso y ventaja de la Arquitectura de Software y como esta se puede representar usando el lenguaje de modelado UML, luego se introducirá los conceptos, clasificación y utilidad de los patrones y la forma en que pueden encadenarse formando Lenguajes de Patrones, estos lenguajes de patrones a su vez pueden marcar la distribución y comportamiento de los componentes de la arquitectura desarrollada en un proyecto de construcción de software, para reusar practicas exitosas de diseño y con ello garantizar la calidad del software.

Arquitectura de Software

Se puede comenzar con la definición de Clements (1998) donde describe que la Arquitectura de Software es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se la percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema. Esta definición permite entender a la Arquitectura de Software como la descripción de un sistema macro, basándose en los componentes o elementos que la integran y como estos se relacionan e interactúan entre sí, es una visión general del sistema con un alto nivel de abstracción que le permite al ingeniero visualizar a la solución general al problema.

Otra de las definiciones de la Arquitectura de Software, es la realizada por Bass (1998) en la cual este autor se refiere a ella como la estructura a grandes rasgos del sistema, estructura consistente en componentes y relaciones entre ellos. Esta estructura se vincula con el diseño, pues la Arquitectura de Software es después de todo una forma de diseño de software que se manifiesta tempranamente en el proceso de creación de un sistema; pero este diseño ocurre a un nivel más abstracto que el de los algoritmos y las estructuras de datos. En el que muchos consideran un ensayo seminal de la disciplina, Garlan y Shaw (1994), sugieren que dicha estructura incluye una organización a grandes rasgos y control global; protocolos para la comunicación, la sincronización y el acceso a datos; la asignación de funcionalidad a elementos del diseño; la distribución física; la composición de los elementos de diseño; escalabilidad y rendimiento; y la selección entre alternativas de diseño, es por ello que representa un diseño ajustable o cambiante hasta que proporcione la mejor solución del problema, y una visión general del sistema antes de que este sea desarrollado.

Garlan (2000), establece en una definición tal vez bastante amplia que la Arquitectura de Software constituye un puente entre el requerimiento y el código, ocupando el lugar que en los gráficos antiguos se reservaba para el diseño. La definición oficial de Arquitectura de Software proporcionada por IEEE (2000), adoptada también por Microsoft, dice así: La Arquitectura de Software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución.

En conclusión, se puede observar que todas las definiciones de la Arquitectura de Software radican en que es una estructura del sistema, representada o descrita a través de sus componentes y como estos se relacionan entre sí.

Para representar una arquitectura de software existen distintos enfoque y perspectiva, en todas es común la presencia de múltiples diagramas y vistas para representar las distintas perspectivas del sistema a desarrollar, al análisis de todos en

conjunto permite al equipo de desarrollo una visión holística del sistema que representa, estos a su vez sirven de medio de comunicación entre los miembros del equipo y como un elemento común para discusión y toma de decisiones.

Existen unos cuantos organismos de estándares (ISO, CEN, IEEE, OMG) que han codificado diferentes aspectos de la Arquitectura de Software, con el objetivo de homogeneizar la terminología, los modelos y los procedimientos. Como producto de este trabajo se han construido marcos de trabajo que reconocen entre tres y seis vistas diferentes del sistema.

Uno de los marcos para representar arquitectura mas empleado es el llamado modelo "4+1", vinculado al Rational Unified Process (RUP), que según Kruchten (1995) define cuatro vistas diferentes de la arquitectura de software: (1) La vista lógica, que comprende las abstracciones fundamentales del sistema a partir del dominio de problemas. (2) La vista de proceso: el conjunto de procesos de ejecución independiente a partir de las abstracciones anteriores. (3) La vista física: un mapeado del software sobre el hardware. (4) La vista de desarrollo: la organización estática de módulos en el entorno de desarrollo. El quinto elemento considera todos los anteriores en el contexto de casos de uso. Lo que académicamente se define como Arquitectura de Software concierne a las dos primeras vistas. El modelo 4+1 se percibe hoy como un intento se reformular una arquitectura estructural y descriptiva en términos de objeto y de UML.

Otro marco de trabajo para representar arquitectura construido sobre la base de UML y confeccionado para usarse en el desarrollo de software de gran escala, es el propuesto por Garland(2003), aquí se propone el uso de tres vistas, una encargada de mostrar los aspectos conceptuales y de análisis de la aplicación, otra dedicada a la representación del diseño lógico y una tercera responsable del ambiente de funcionamiento y despliegue, en las Tabla 1 se puede apreciar los entregables y componentes en términos de diagramas de cada una de ellas.

	Nombre	Diagrama UML	Descripción
Conceptual y Análisis	Centrado en el Análisis	Clases	Describe las entidades del Sistema en respuesta a un escenario, frecuentemente se refiere a una vista de las clases participantes
	Análisis de Interacción	Interacción	Diagrama de Interacción entre objetos del análisis
	Análisis General	Clases	Combinación de todas las clases detectadas en las distintas vistas que se centran al análisis de un escenario en particular.
	Contexto	Casos de Uso	Muestra los sistemas externos, actores y el propio sistema bajo diseño
Diseño Lógico	Componentes	Componente	Ilustra la comunicación entre los diversos componentes
	Interacción de Componentes	Interacción	Muestra la Interacción entre los diversos componentes
	Estado de Componentes	Estado/Actividad	Diagrama de Estado u/o actividad de un componente o conjunto de componentes.
	Capas de Subsistemas	Paquetes	Ilustra las capas y los subsistemas.
	Datos Lógicos	Clases	Muestra una vista de los datos críticos sensibles para la integración.
	Interfaces de Dependencia entre Subsistemas	Clases	Muestra la dependencia de los subsistemas y sus diversas interfaces.
Contexto y Despliegue	Despliegue	Despliegue	Mapea el software y su ubicación en los distintos dispositivos de hardware.
	Datos Físicos	Despliegue	Vista física de una base de datos en particular.
	Procesos	Despliegue	Muestra los procesos de una instancia particular de sistema.
	Estado de Procesos	Estado	Muestra los estados dinámicos de un procesos

Tabla 1. Vistas de una Arquitectura de Software.
Fuente: El Autor.

Para efectos de este trabajo se utilizara este segundo método de representación de arquitectura, pues muestra de una forma intuitiva y concreta al equipo de trabajo los componentes estructurales de la aplicación; Garland(2003) afirma que desde sus comienzos fue concebida como una forma ágil de representar arquitectura, exponiendo

que el arquitecto puede hacer uso de aquellos artefactos que verdaderamente presenten información útil al equipo de diseñadores.

Patrón de Software

En el sentido más general, un patrón es un modelo a seguir que describe una solución exitosa de un problema particular en un contexto dado.

Zambrano y Acosta (2001) se refieren a un patrón desde el punto de vista informático, como una descripción de una solución computacional exitosa a un problema recurrente en un contexto particular; esta solución expresa la experiencia y el conocimiento de expertos y, en este sentido, sirve como medio de comunicación para difundir este conocimiento.

Un patrón soluciona un problema recurrente, ya que si el problema es poco o nada frecuente, entonces pudiera no tener sentido construir un patrón de su solución, el cual no se utilizará más. De allí que, una de las ventajas de los patrones es fomentar y facilitar el reúso, en este caso, de soluciones.

Los patrones facilitan la reutilización de diseños y arquitecturas de software que han tenido éxito, además capturan la experiencia y la hacen accesible a los no expertos, se pueden agrupar y el conjunto de sus nombres forma un vocabulario que ayuda a que los desarrolladores se comuniquen mejor. Ayudan a los desarrolladores a comprender un sistema más rápidamente cuando está documentado con los patrones que usa. Facilitan la reestructuración de un sistema tanto si fue o no concebido con patrones en mente. Pavón (2004)

Los patrones según Buschmann (1996) le ayudan a construir sobre la experiencia colectiva de ingenieros de software experimentados. Estos capturan la experiencia existente y que ha demostrado ser exitosa en el desarrollo de software, y ayudan a promover las buenas prácticas de diseño. Cada patrón aborda un problema específico y recurrente en el diseño o implementación de un sistema de software.

Para que un patrón sea efectivo debe según Buschmann (1996):

- **Resolver un problema**: Los patrones capturan soluciones, no principios o estrategias abstractas.
- **Ser concepto probado**: Capturan soluciones, no teorías o especulaciones.
- **Describir una relación**: Los patrones no describen módulos sino estructuras y mecanismos.
- **Poseer un componente humano significante**: El software sirve a las personas. Los mejores patrones aplican a la estética y a las utilidades.

Clasificación de los Patrones

Buschmann (1996), define una taxonomía para los patrones, el cual los reúne en tres grandes grupos, estos en cada grupo varían respecto a su nivel de detalle y abstracción esto se puede ver en la Figura 3:

Figura 3. Patrones según el nivel de detalle, adaptado de POSA.
Fuente: Microsoft(2004)

Patrones de Arquitectura ó Arquitecturales

Estos expresan un paradigma fundamental para estructurar un sistema software y proporcionan un conjunto de subsistemas predefinidos, que especifica sus responsabilidades, e incluye reglas y guías para organizar las relaciones entre ellos. Algunos ejemplos: Tuberías y filtros, Cliente/Servidor, Maestro-Esclavo, Control centralizado y distribuido

Patrones de diseño

Los patrones de diseño, están compuestos de varias unidades arquitecturales más pequeñas y describen el esquema básico para estructurar subsistemas y componentes resolviendo un diseño general dentro de un contexto particular. Como por ejemplo: Proxies, Factorías, Adaptadores, Composición, Broker.

Patrones elementales (idiomas)

Un patrón de idiomas, es un patrón de bajo nivel, específico a un lenguaje de programación. Un patrón de idiomas describe cómo llevar a cabo aspectos particulares de componentes o las relaciones entre ellos, usando las características de lenguaje de programación utilizado. Por ejemplo: Modularidad, Interfaces mínimas, Encapsulación, Objetos, Acciones y Eventos, Concurrencia.

Lenguaje de Patrones

Los patrones se pueden agrupar formando colecciones que constituyen un vocabulario para comprender y comunicar ideas. Cuando estas colecciones se conforman con habilidad para formar un todo cohesionado que revele las estructuras y las relaciones de sus componentes para cumplir un objetivo compartido, entonces se esta hablando de lo que Alexander (1977) en el urbanismo bautiza como lenguaje de patrones. Un lenguaje de patrones define una colección de patrones, reglas y pautas para combinarlos con un estilo que se podría denominar arquitectónico, pues define una forma de construir estructuras a todos los niveles de escala y en todos los grados de diversidad. En realidad, un lenguaje de patrones se compone de un léxico de patrones y una gramática que establece cómo unirlos para formar estructuras sintácticas. Idealmente, los buenos lenguajes de patrones son generativos, es decir,

capaces de generar todas las posibles frases a partir de un vocabulario de patrones rico y expresivo.

Lenguaje de Patrones para Aplicaciones Empresariales

Fowler(2003) hace una recopilación muy completa de un conjunto de patrones de Arquitectura y Diseño que pueden emplearse en la construcción de aplicaciones empresariales, estos se pueden enlazar de manera coherente en la forma de lenguaje de patrones, para que el arquitecto establezca un contrato de comportamiento para los componentes de la Arquitectura y la forma en que se comunican.

Para el desarrollo del presente trabajo el autor selecciono un conjunto de patrones de esta colección y la agrupo para generar un esquema de funcionamiento que puede aplicarse a cualquier aplicación empresarial, incluyendo la abordada en el presente trabajo, esto puede verificarse en la Figura 4.

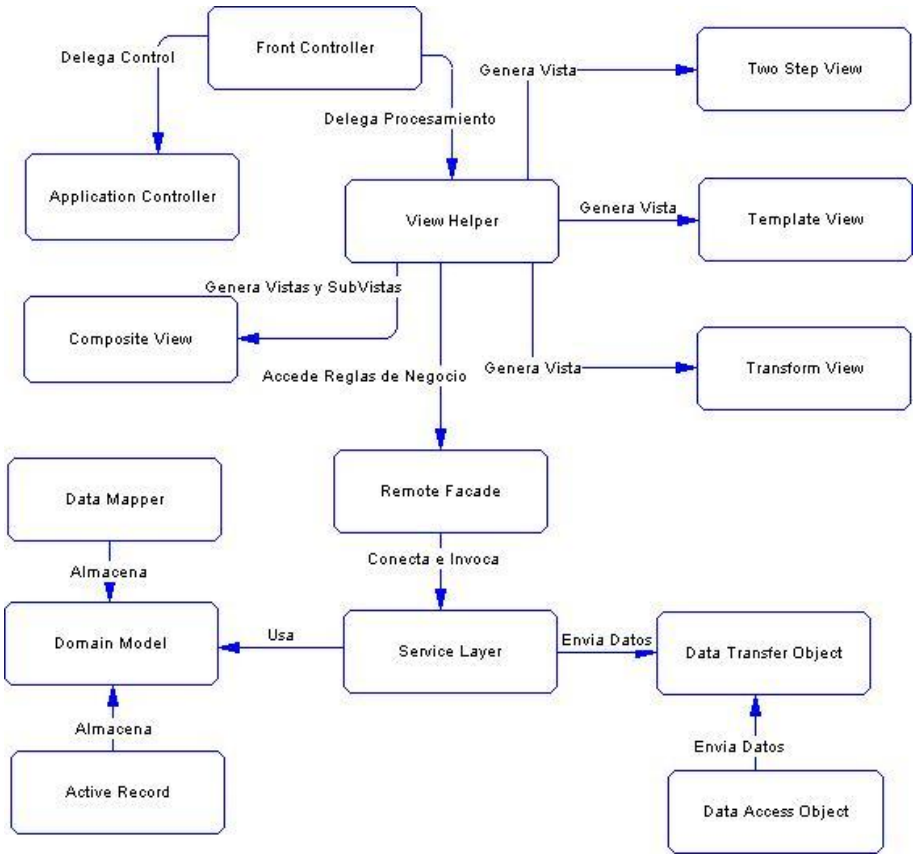

Figura 4. Lenguaje de Patrones para Aplicaciones Empresariales.
Fuente: El Autor.

En este sentido se puede decir que una aplicación debe estar marcada por una división en capaz, estas son:

- Presentación. Ofrece los servicios necesarios para la presentación de información.
- Lógica del Negocio. Encapsula la lógica relacionada a la situación del mundo real que modela la aplicación.
- Fuente de Datos. Proporciona la comunicación con Bases de Datos, Sistemas de Mensajería, Archivos XML y Otros Sistemas.

La transferencia de los datos entre las Capas de la Aplicación se hará usando el patrón Data Transfer Object.

La Capa de Lógica de Negocio estará dividida en una Capa construida bajo el Patrón Service Layer y otra que usa el patrón Domain Model, así mismo se desarrollo una capa entre ambas direccionada por el patrón Data Access Object.

En la Capa de fuente de Datos se empleara el Patrón Data Mapper (Data Mapper).

La Capa de Presentación usa los servicios proporcionados por la Lógica de Negocio enviando mensajes a la implementación de Service Layer. En la Capa de Presentación se emplean los siguientes patrones:

- Composite View.
- Application Controller.
- Two Step View.
- Template View.
- View Helper.
- Transform View.
- Front Controller.

Con miras a garantizar que la Lógica del Negocio pueda reusarse desde otros nodos u aplicaciones, se incorporara el patrón Remote Facade, esto es particularmente importante ahora que se piensa en una plataforma interoperable para que la lógica de negocio de la aplicación sea reusada por otras aplicaciones.

Hay patrones complementarios que servirán para direccionar el diseño de la Capa Lógica de Negocio, estos son:

- Gateway Layer
- Supertype.
- Separated Interface.
- Service Stub

DESARROLLO DE LA PROPUESTA

Los problemas detectados en las sesiones JAD realizadas en conjunto con el personal responsable de los distintos Registros Académicos de la UCLA y un trabajo de análisis documental sobre software para control de estudio y registro académico disponible en el mercado, permitieron establecer los requerimientos funcionales y no funcionales del sistema que requiere UCLA para mejorar su desempeño en los procesos de gestión académica y docente actuales, esto como un paso previo a la construcción de la arquitectura que soportara el software con el que se puede dar una solución automatizada a la situación.

Requisitos Funcionales

La UCLA está constituida por Decanatos, los cuales son órganos administrativos y académicos a través de los cuales, la Universidad realiza sus funciones de docencia, investigación y extensión. Para que en los Decanato se puedan llevar a cabo estas actividades, se requiere de la administración de recursos tanto físicos (aulas, laboratorios, equipos, etc.), humanos (profesores, auxiliares docente, personal administrativo, obreros) como de tiempo. Es importante acotar que tanto a Nivel universitario como a nivel de cada Decanato existe un marco legal que controla el desarrollo de las actividades académicas, docentes y de investigación. (Ver Figura 5)

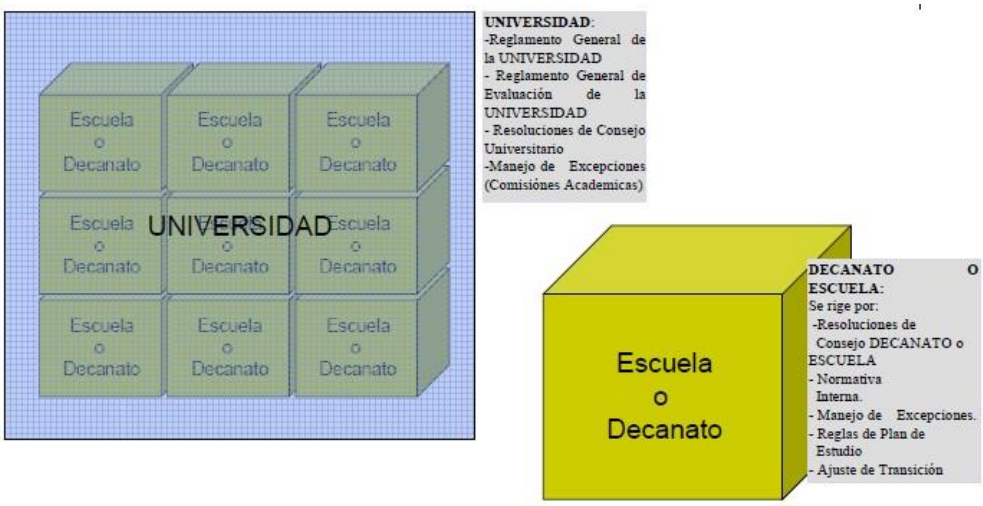

Figura 5. Esquema Universidad, Decanatos y Marco Legal.
Fuente: El Autor.

Los Decanatos a su vez están conformados por Programas, Departamentos y demás dependencias de carácter académico y administrativo. Las labores docentes de cada Decanato, serán realizadas a través de los Programas que la integran. Por su especial naturaleza a cada Programa le corresponde enseñar e investigar un grupo de disciplinas fundamentales y afines, dentro de una rama de la ciencia y la cultura.

El ingreso de los estudiantes a UCLA es controlado por la Dirección de Admisión y Control de Estudio y es la primera instancia que se visita para que cualquier persona interesada en cursar algunos de los programas académicos aperture su expediente, en este proceso se recoge un grupo de información de gran importancia e interés para otras direcciones de la Universidad, para corporando respectivamente información personal, datos socioeconómicos, entre otros.

Este grupo de disciplinas conforman lo que se llama Plan de Estudios y es el que debe ser cursado y aprobado por un estudiante, para así cumplir con los requisitos académicos y obtener el título conferido por dicha carrera universitaria. A su vez el plan de estudios es dictado durante un lapso académico, que viene a ser el período

educativo cuya duración puede ser semestral, anual o de 5 semanas para un período Extraordinario o Intensivo. Existen decanatos cuya duración es mixta, por lo que manejan tanto el régimen semestral como anual. Asimismo, en algunos decanatos puede haber varios planes de estudios abiertos a la vez. (Ver Figura 6)

Cualquiera que sea el caso, semestral, anual o mixto, se realizan una serie de actividades o procesos que pueden ser agrupados por etapas, estas son: Planificación, Ejecución y Finalización. Por lo tanto el funcionamiento del Sistema de Gestión Académica CUM LAUDE, girará en torno al desarrollo de un Lapso Académico.

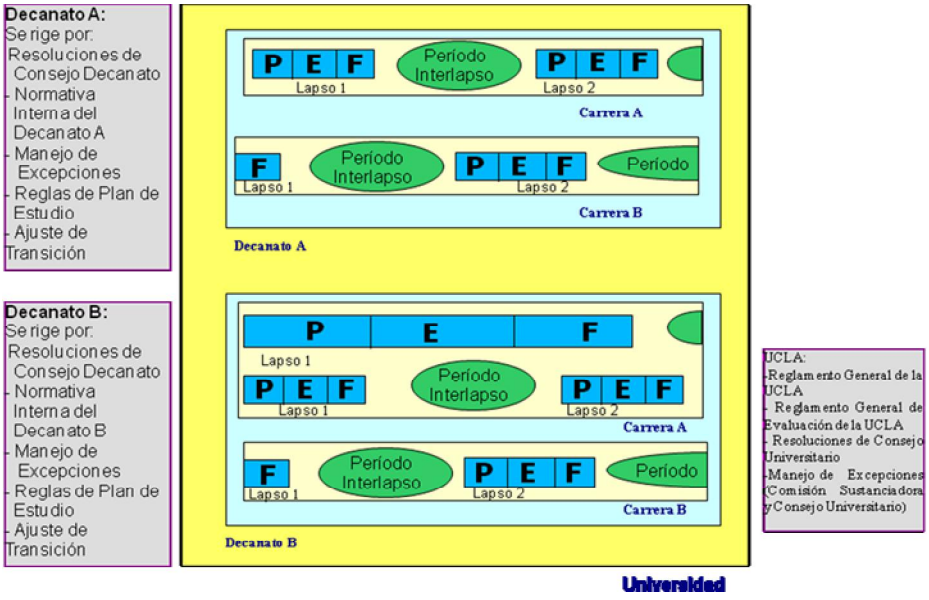

Figura 6. Esquema sobre Universidad, Decanatos y Reglamentos.
Fuente: El Autor.

La planificación de un lapso académico se inicia con la Creación del Lapso Académico, en donde se definirá el lapso académico, el tipo de lapso y la fecha de inicio del mismo. Dependiendo del tipo de lapso seleccionado, se generarán las actividades que se llevarán a cabo durante el mismo. Una vez definido el Lapso

Académico, se puede consultar como queda estructurado el Calendario Académico con todas sus actividades, tales como: el inicio de clases, inclusión y retiro de asignatura, cancelación de semestre, inscripción a la prueba extraordinaria, presentación de prueba extraordinaria, finalización de semestre, así como cualquier otra actividad que sea aprobada por el Consejo de Decanato. El sistema permitirá realizar la Reprogramación de un Lapso Académico previa aprobación de Consejo de Decanato y Consejo Universitario. Con está información se establece el Calendario Académico para un Lapso determinado.

Otro de los procesos que se lleva a cabo en esta etapa es la Creación de la Oferta Académica, esta opción permitirá establecer el número de secciones que se ofertarán en el lapso académico que está por iniciar, así como también permitirá realizar consultas sobre la oferta académica realizada en los lapsos anteriores, lo que ayudará al usuario en la toma de decisiones para el lapso que va a iniciar. En esta opción el usuario podrá estimar el número de secciones que se van abrir para el lapso académico, una vez que haya sido aprobada la Oferta Académica.

Una vez aprobada la oferta académica, se llevará acabo el proceso de Administración de Secciones, esta opción permitirá al usuario actualizar y distribuir las asignaturas por sección para un Plan de Estudios en un Lapso Académico determinado.

En esta etapa del sistema se llevará a cabo la asignación de la Carga Académica del Docente para el Lapso Académico que está por iniciar.

La Configuración de Horarios consiste en distribuir las asignaturas ofertadas por secciones, en un día y hora determinada junto con los recursos necesarios de acuerdo a la naturaleza de la asignatura, estos pueden ser físicos, humanos y de tiempo.

Otra de las actividades que se ejecutan en esta etapa de Planificación es la Configuración de Grupos de Inscripción, Esta opción permite crear los grupos a los que le serán asignados los criterios de inscripción, a cada grupo se le establece la fecha, hora de inicio y finalización que le corresponde para la inscripción, una vez que se han definido los grupos, se realiza el Manejo de Grupos de Inscripción a través del cual se le cambia el estatus a cada uno de los grupos de inscripción, es decir en espera, inscribiendo o cerrado.

Una vez culminada la etapa de planificación, el sistema da inicio a una nueva etapa, la cual llamaremos Etapa de Ejecución. En esta etapa encontramos el proceso Inscribir Alumnos que permitirá a un estudiante registrar la carga académica que le corresponde para ese lapso académico de acuerdo al plan de estudios que este asociado a él.

Concluido el proceso de inscripción e iniciada las clases, comienzan otros procesos, entre los que se encuentra la Inclusión y Retiro de Asignaturas, los cuales permitirán modificar la carga académica de un estudiante, ya sea agregando una nueva asignatura a su carga académica o eliminando alguna de las asignaturas ya inscritas.

El estudiante también podrá realizar la Cancelación de Matrícula que no es más que anular su carga académica para el lapso que se encuentra activo.

Otro de los procesos que se lleva a cabo durante la etapa Ejecución es el Cambio de Secciones, se puede realizar de dos formas, el cambio mutuo de secciones, que consiste en realizar el cambio de mutuo acuerdo entre los estudiantes en la misma asignatura pero en secciones diferentes, o cambiar al estudiante a otra sección pero con la misma asignatura, este proceso se llevará a cabo previa autorización.

El sistema también permitirá llevar a cabo el registro de los estudiantes para presentar las Pruebas Extraordinarias, en aquellas asignaturas que hayan inscrito

previamente o Cursar asignatura bajo Régimen Especial, en ambos procesos se facilitará el envío de la información requerida a los departamentos correspondientes y el registro de las calificaciones obtenidas por los estudiantes.

De igual manera, en esta etapa encontramos la opción Actividades de Coordinación de Asignaturas, en la cual se planificarán y registrarán las actividades que llevarán a cabo los docentes durante el lapso académico, tales como la realización de evaluaciones, registro de las calificaciones para cada parcial, entrega de proyectos, entre otras.

A medida que se vayan cumpliendo las actividades de evaluación a los estudiantes, el docente tendrá la oportunidad de registrar las calificaciones obtenidas de las evaluaciones realizadas, esto se hará en base a la planificación definida por la Coordinación de Asignatura, el proceso se llevará a cabo a través de la opción de Trascribir Calificaciones por Sección. Igualmente el docente podrá Modificar la Nota Final de un estudiante en una asignatura una vez concluido el lapso académico.

También a través del sistema se podrá verificar aquellos estudiantes que realicen solicitud de reingreso, el sistema emitirá información acerca de si le corresponde o no al estudiante reingresar para su activación en el próximo lapso académico.

Una vez concluidas las actividades planificadas en el Calendario Académico se procede a dar inicio a la etapa de finalización, en la que se encuentra el proceso Cierre de Lapso, que a su vez está conformado por una serie de actividades, entre las que tenemos Generar Boletines. Una vez cerrado el proceso de Transcribir Calificaciones por Sección, se pueden generar los boletines ya sea en lote o individual, los boletines contendrán la actuación académica del estudiante de acuerdo a la carga académica que curso en el lapso.

Una vez que se ejecuta el Cierre de Lapso se realiza la Emisión de Boletines, que consistirá en mostrar la información académica de cada estudiante, su índice académico, prioridad de inscripción, asignaturas que puede cursar el próximo lapso académico y mostrará información en caso de que no este solvente por Biblioteca o Librería Rental, la información podrá ser consultada a través de Internet o enviada vía web a los correos de los estudiantes o imprimiendo el boletín.

Al finalizar el proceso de Generar Boletines, el sistema producirá los listados de:

- Posibles estudiantes que serán sancionados por RR o RP en el próximo lapso académico.
- Los estudiantes que fueron sancionados por RR o RP para el próximo lapso académico.
- Los estudiantes que solicitaron reingreso para el próximo lapso académico luego de cumplir sanción por RR o RP.

Una vez concluidos los procesos de la etapa de Finalización, se iniciarán una serie de actividades que se llevan a cabo entre la etapa de finalización y la etapa de inicio del nuevo lapso académico, esta etapa la llamaremos Inter-Lapso.

Durante esta etapa se realiza la Apelación por parte del estudiante ante la Comisión Sustanciadora. Una vez que el estudiante ha sido sancionado por RR o RP, puede recurrir ante la Comisión Sustanciadora para que le sea suspendida la sanción. El sistema permitirá registrar la solicitud de apelación de la sanción y el veredicto que se emita, asimismo se podrá consultar el número de apelaciones realizadas por el estudiante y las respuestas emitidas por la Comisión.

Otro de los procesos que se realizan en la etapa de Inter-Lapso es la Preinscripción de los estudiantes para el próximo Lapso Académico, ya sea para un lapso normal o intensivo, esto permitirá conocer la demanda que tendrán las asignaturas, en el período académico a iniciar.

El Sistema de Información para la Gestión Académica CUM LAUDE, además de realizar el conjunto de procesos expuestos anteriormente, podrá llevar a cabo una serie de actividades que estarán disponibles para ser ejecutados cuando sea requerido por el usuario. Entre estas actividades se encuentra Generar Plan de Estudios que consiste en registrar los nuevos planes de estudios que son aprobados para un programa de un Decanato, allí se registrarán las asignaturas que conformarán el nuevo plan de estudios, el código de la asignatura, el semestre o año al que corresponde, la densidad horaria, las unidades de crédito y las prelaciones.

Del mismo modo, el sistema permitirá registrar las Equivalencias, estas pueden ser: Equivalencias Internas o Externas, este proceso consiste en registrar el veredicto que otorga el Consejo Universitario de la UCLA, previa revisión y análisis de los entes correspondiente, en este veredicto se indican las asignaturas que le fueron aprobadas al solicitante.

Equivalencias entre Planes de Estudios o Cambio de Pensum, está se da cuando en un mismo programa hay varios planes de estudios abiertos y se quiere realizar el proceso de transición de un plan de estudios a otro, ya sea porque se va a cerrar uno de los planes de estudios o porque el estudiante decide realizar la transición al nuevo plan de estudios.

El sistema ofrecerá una serie de servicios los cuales pueden ser solicitados en cualquier momento de la ejecución del lapso académico, alguno de los servicio ofrecidos son: Constancias de Estudios, inscripción, de Notas, historial, de culminación de carrera, de solicitud de anteproyecto del trabajo especial de grado, de buena conducta, retiro definitivo, de estar incurso en el articulo 35 literal (a) o (b), Carta de culminación, entre otras. También emitirá estadísticas: Alumnos inscritos (matrícula), regulares y repitientes por asignatura, rendimiento académico por lapso,

lista de carga académica, inscritos por semestre, matrícula por edad y sexo, entre otras.

Otra de las opciones que puede realizarse, es la Admisión de los estudiantes Nuevo Ingreso, está se ejecutará desde la Dirección de Admisión y Control de Estudios de la UCLA, el estudiante deberá estar preinscrito previamente por medio de la página Web de la UCLA. En la opción de Admisión, se completarán los datos del estudiante que ha sido aceptado para ingresar a cursar alguno de los programas dictados en la Universidad y se registrarán los documentos que fueron consignados al momento de su admisión.

El sistema también permitirá llevar un Registro de los Recursos Físicos y Humanos que serán necesarios durante la Configuración de Horarios. Se registrará la capacidad, características y descripción de los recursos físicos y se podrá consultar la disponibilidad de los mismos durante un lapso académico. Del personal se registrará sus datos básicos y laborales así como poder consultar su disponibilidad durante un lapso académico.

Asimismo, el sistema manejará la opción Gestión Estudiante que permitirá actualizar y consultar los datos del estudiante, realizar el Retiro Definitivo de un estudiante de la Universidad, cancelando las asignaturas que se encontraba cursando para el momento de la solicitud y retirándolo del programa para el que estaba inscrito. También se podrán Registrar las Equivalencias de aquellos estudiantes que ingresen bajo esta modalidad, primero es registrado como un estudiante nuevo ingreso y previo a la generación de su carga académica, se registran las asignaturas que fueron aprobadas por la Comisión de Equivalencia. Si el estudiante incurre en alguna falta, de acuerdo a las establecidas en los Reglamentos de la UCLA, se registrará por sistema la Sanción Disciplinaria aplicada al estudiante, el tipo de falta y la sanción que deberá cumplir el estudiante.

Se podrá realizar la Configuración de los Decanatos esta opción permite al usuario registrar las Autoridades de un Decanato, tales como: Decano, Directores de Programa, Jefes de Departamento, y Coordinadores existentes. El sistema ofrecerá un asistente de configuración, que le facilitará al usuario llevar a cabo el registro de la información.

Requisitos no Funcionales

El análisis de necesidades recogidas en los diversos registros académicos de la UCLA, arrojo que existe una variedad de dinámicas de trabajo entre los distintos decanatos, incluyendo importantes diferencias en interpretación del reglamento, organización de los procesos de inscripción, inclusión y retiro de asignatura, transcripción de notas parciales y finales, entre otros, así mismo existen diversos perfiles de usuarios, con variadas expectativas de información. Todos estos elementos condicionan las características no funcionales del software que se requiere, en este sentido el software debe sugiere que este sea primordialmente:

- Mantenible, Para que pueda adaptarse fácilmente y en el menor tiempo posible a la dinámica que cada decanato maneja.
- Confiable, Dada lo critico y sensible de la información que maneja es importante ofrecer al usuario la sensación de que se está manejando procesos confiables en funcionamiento y coherencia.
- Usable, Dada la gran cantidad de usuario que podrá utilizarlo es necesario que su filosofía de trabajo tanto para los administradores como para usuarios finales sea de fácil aprendizaje y manejo.
- Interoperable, La universidad incluye polos de trabajo relacionados a la gestión administrativa a los que debe integrarse el sistema, de manera rápida y sencilla.

Lista de Funcionalidades

Mediante un proceso de inspección de los requisitos podemos establecer la siguiente lista de funcionalidades. (Ver Tabla 2)

Nombre de la Funcionalidad	Usuario Responsable
Gestión de Lapso Académico. • Definir Tipos de Lapso Académico. • Definir Lapso Académico y Calendario Académico Correspondiente.	Director de Programa
Definición de Oferta Académica. • Estimar Oferta Académica (Indicadores, Proyección del Lapso anterior, cantidad y capacidad de las secciones, tomar en cuenta la cantidad de grupos que tendrá la sección y su capacidad). • Aprobar Oferta Académica.	Director de Programa
Conformación de Secciones. • Crear o cerrar secciones (se realizará después de aprobada la oferta académica, durante el proceso de inscripción o inclusión).	Director de Programa
Asignación Carga Académica Docente.	Jefe de Departamento
Configuración Horarios para los Recursos Físicos y Humanos. • Aprobar horarios.	Director de Programas/ Jefes de Departamentos
Configuración de Proceso de Inscripción. • Grupos de Inscripción. • Criterios de Inscripción.	Administrador del Sistema
Inscripción de Estudiante. Casos: Nuevo Ingreso, Regulares, Intensivos. • Parámetros: Reglamentos Generales, Criterios de Prosecución, Solvencia de Biblioteca. • Validaciones: Chequeo de horarios, Carga Máxima. • Disponibilidad de la sección. • Censo de la demanda no satisfecha.	Estudiante y / o Registro Académico
Inclusión y Retiro de Asignatura. • Excepciones: Cuando la solicitud se efectúa fuera de lo planificado.	Estudiante y / o Registro Académico
Cancelación de Matrícula. • Excepciones: Cuando la solicitud se efectúa fuera de lo planificado.	Registro Académico
Retiro Definitivo de Estudiante.	Registro Académico
Servicios (Constancias, Solvencias, Records Académicos).	Estudiante, Profesor, Registro Académico
Transcripción Notas por Sección. • Registrar resultados de evaluaciones planificadas. • Generar cortes parciales (estadísticas). • Finalizar proceso de trascripción. • Manejo de trascripciones sin cortes parciales.	Profesor
Corrección de Nota Final.	Profesor
Gestión de Prueba Extraordinaria y Otro Régimen. • Registrar Solicitud. • Transcripción de Calificación (Resultado del Régimen podrá afectar el Record Académico).	Jefe de Departamento, Registro Académico
Tesis de Grado.	Registro Académico
Pasantías.	Registro Académico
Trabajo Comunitario.	Registro Académico
Solicitud de Reingreso/Reincorporación.	Registro Académico / Estudiante
Actividades de Coordinación de Asignaturas. • Definir Coordinaciones. • Planificar Actividades (Evaluaciones). • Registrar Calificación por Coordinación. • Definir Escala de Evaluación.	Jefe de Departamento / Coordinador de Area

Cierre de Lapso Académico. • Generar Boletines. • Generar Listado de Estudiantes que reingresan por RR o RP. • Asignar Prioridades de Inscripción. • Generar posibles Sanciones RR y RP. • Generara Sancionados RR y RP. • Estadísticas.	Administrador del Sistema
Comisión Sustanciadora. • Registrar Apelaciones. • Registrar Veredicto de Apelación.	Registro Académico
Preinscripción Lapso Académico.	Estudiante
Gestión de Estudiantes. • Ficha del Estudiante. • Retiro Definitivo. • Registrar Equivalencias del Estudiante. • Sanciones. o Académicas. o Disciplinarias.	Registro Académico
Gestión de Recursos. • Recursos Físicos • Espacio Físico • Recursos • Asignación de un Recurso a un Espacio Físico.	Administrador del Sistema / Registro Académico
Gestión Docente. • Ficha Docente. • Contrataciones. • Ascensos. • Cambios de Dedicación. • Planificación de Actividades.	Administrador del Sistema/Docente
Gestión de Programas. • Definir Plan de Estudios. • Asignaturas. • Equivalencias entre Planes de Estudios. o Equivalencias Internas. o Cambio de Pensum. o Equivalencias Externas.	Director de Programa / Registro Académico
Configuración Decanatos. • Departamentos. • Autoridades.	Administrador del Sistema
Apertura de Expediente Estudiante	Dirección de Admisión y Control de Estudio
Proceso de Ingreso a la Universidad	Dirección de Admisión y Control de Estudio

Tabla 2. Funcionalidades Software de Control de Estudio y Registro Académico.
Fuente: El Autor.

Modelo de Requisitos

A continuación se presenta un conjunto de diagramas de actores y casos de uso, mostrando los requisitos relacionados al sistema de Control de Estudio y Registro Académico necesitado en UCLA. (Ver Figuras 7,8,9,10 y 11)

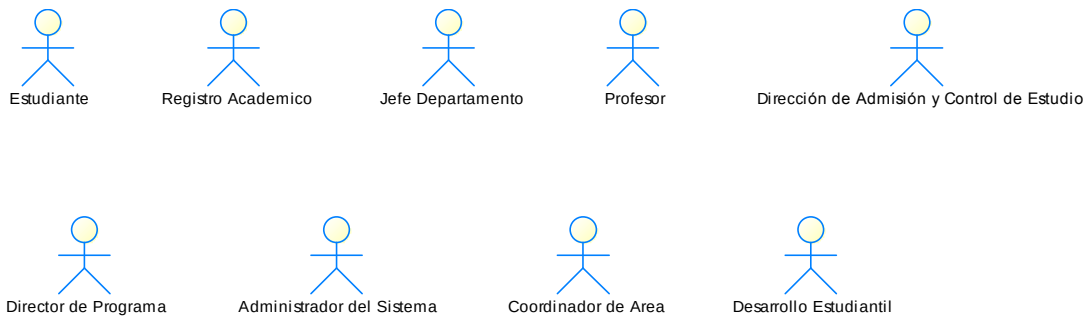

Figura 7. Diagrama de Actores
Fuente: El Autor.

Figura 8. Diagrama de Casos de Uso
Fuente: El Autor.

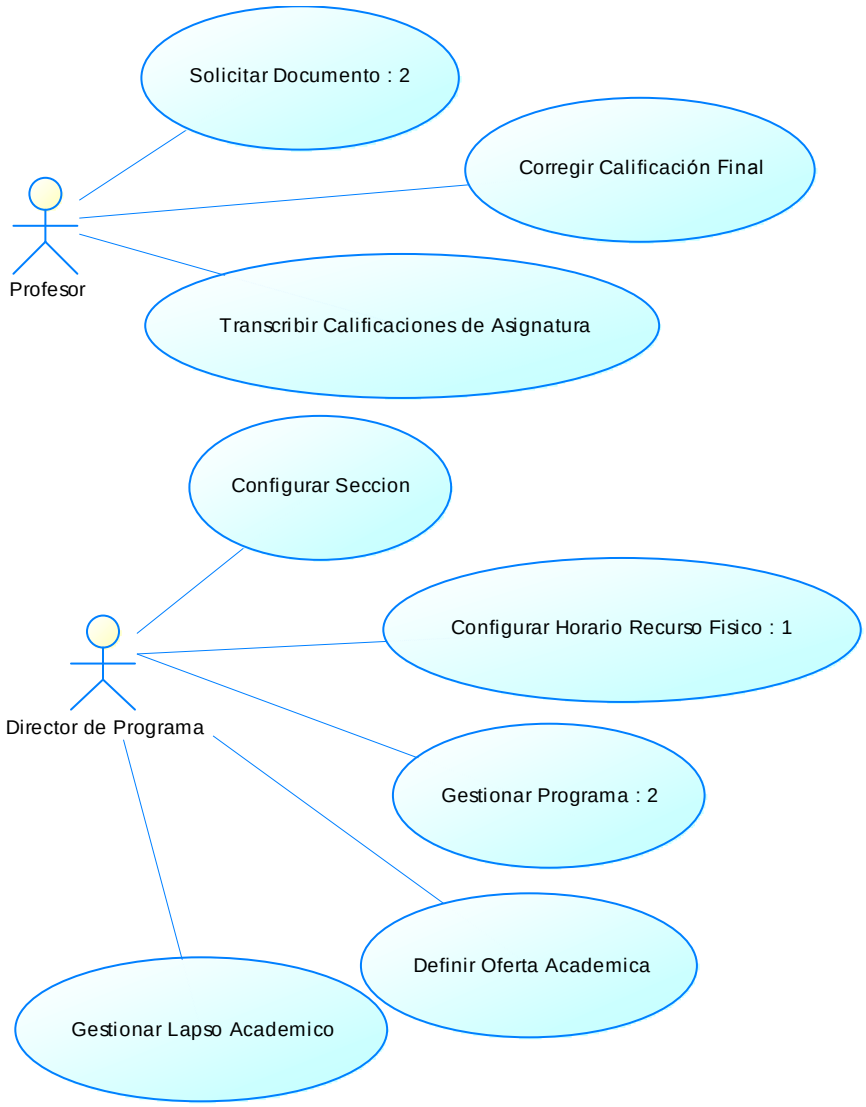

Figura 9. Diagrama de Casos de Uso
Fuente: El Autor.

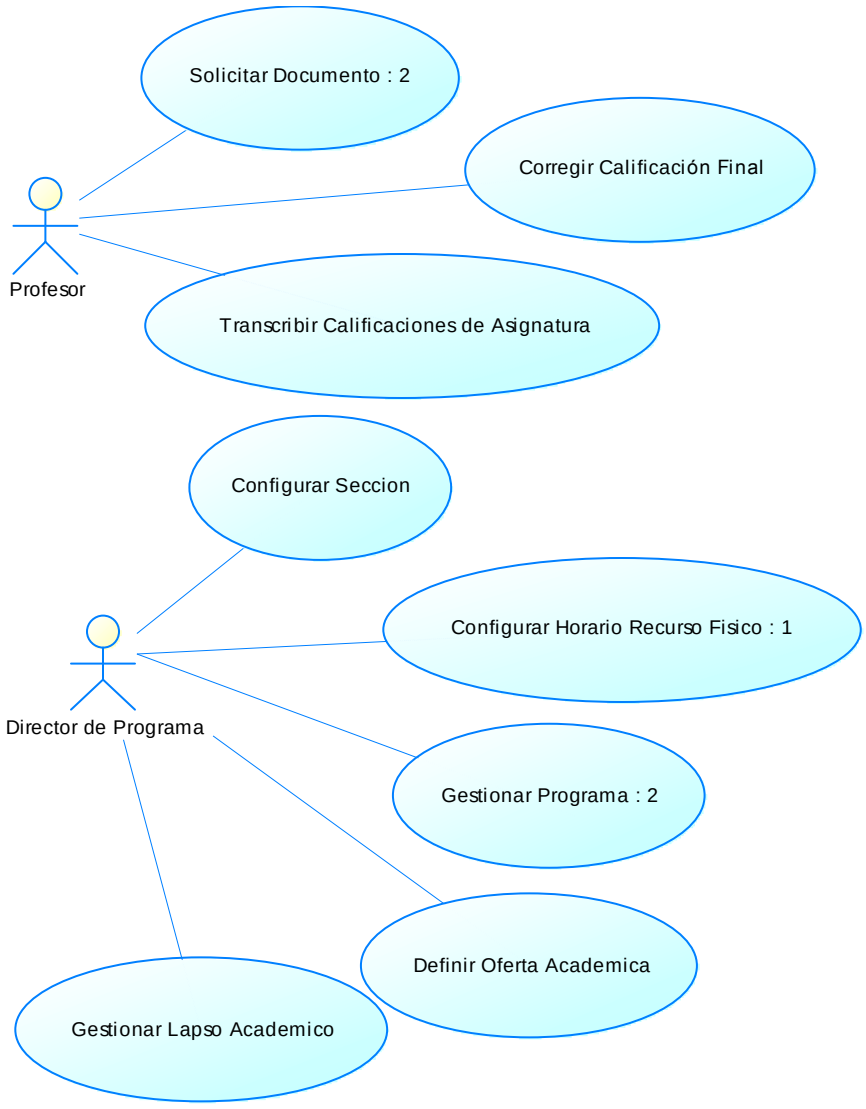

Figura 10. Diagrama de Casos de Uso
Fuente: El Autor.

Figura 11. Diagrama de Casos de Uso
Fuente: El Autor.

Propuesta de Arquitectura de Software

Para definir la arquitectura de software correspondiente al sistema de Control de Estudio y Registro Académico en UCLA, se emplearan algunas de las vistas propuesta por Garland(2003), en especifico aquellas que para este caso de estudio resulten de valor y relevancia para alcanzar los objetivos propuestos.

En este caso el autor se propone a dar una visión general de cómo el sistema está ubicado dentro de la organización y con qué dependencias o actores trabaja, debe darse prioridad a la incorporación de patrones de diseño que direccionen un desarrollo orientado a servicios para garantizar su interoperabilidad y acceso a la información.

Es indispensable igualmente dar un panorama de todos los componentes que conformaran el sistema, como ellos se conectan y se despliegan en la infraestructura de hardware, con el fin de:

- Establecer la división de responsabilidades y roles dentro del equipo.
- Marcar puntos de discusión acerca de la pertinencia desde el punto de vista arquitectónico de la ubicación física de los componentes de software.
- Establecer los equipos y la infraestructura de red necesaria para el desarrollo e implementación de la solución de software.

Desde el punto de vista conceptual se considerara representar una visión de las entidades y conceptos involucrados en el dominio del problema, como un insumo que pueda completarse en futuras iteraciones del proceso de desarrollo.

A partir de este momento el autor se referirá al software de Control de Estudio y Registros Academico como CUM LAUDE, con el fin de darle vida al proyecto que

servirá para dinamizar la toma de decisiones en el área académica y docente en la UCLA.

La Figura 12 muestra un diagrama de contexto para entender que actores, entidades y sistemas externos, interactuaran con el sistema CUM LAUDE.

Figura 12. Diagrama de Contexto – Sistema CUM LAUDE
Fuente: El Autor.

Sobre la base del análisis de los distintos casos de uso y funcionalidades presentados anteriormente en relación a CUM LAUDE, se puede indicar que el

software debe estar formado por dos componentes fundamentales, esto se puede evidenciar en la Figura 13.

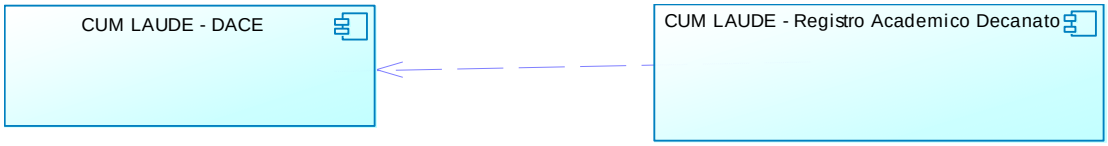

Figura 13. Diagrama de Componentes – Sistema CUM LAUDE

Fuente: El Autor.

Estos componentes se desplegaran en forma de distribuida en dos localidades distintas, uno en la propia Dirección de Admisión y Control de Estudio y otro en cada uno de los decanatos que conforman la UCLA. (Ver Figura 14).

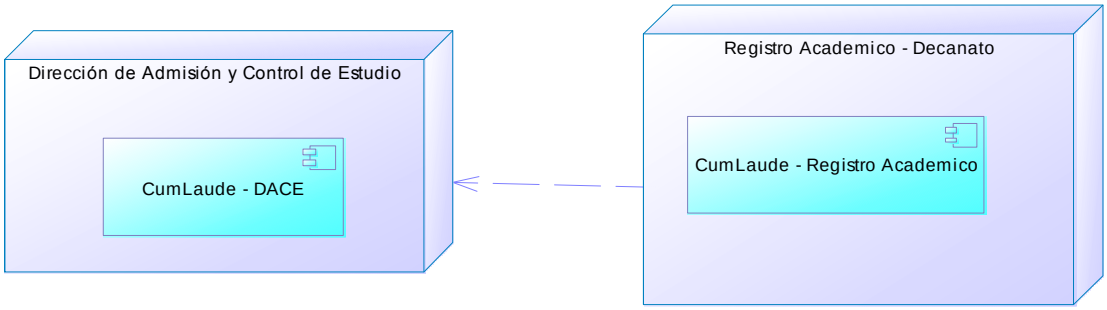

Figura 14. Diagrama de Despliegue - Sistema CUM LAUDE

Fuente: El Autor.

Esta arquitectura distribuida estará formada por:

- Un Componente que residirá en los Registros Académicos de cada uno de los Decanato de la UCLA, capaz para gestionar los procesos académicos que se llevan a cabo en esa localidad, empleando políticas de seguridad, auditoria, respaldo y tolerancia a fallos, de tal manera que se garantice la exactitud y

correctitud de los datos que serán consumidos posteriormente por DACE. Así mismo debe ser lo suficientemente configurable para adaptarse a las necesidades particulares de cada decanato y administrar estudiantes, docentes, programas académicos y recursos administrativos.

- Un componente de software que residirá en la DACE que se encargara de consolidar la información producida en los distintos decanatos, usando para ellos mecanismos abiertos y estandarizados de intercambio de datos. El sistema de información debe estar en capacidad de dar información de un estudiante desde su ingreso hasta la culminación de su plan de estudios, entregando la información pertinente a las diferentes unidades y la requerida para la Acreditación y Auto Evaluación de la UCLA.

La Figura 15 muestra un diseño más detallado de la arquitectura propuesta para CUM LAUDE, allí se puede evidenciar que componentes integran los dos grandes elementos de la aplicación.

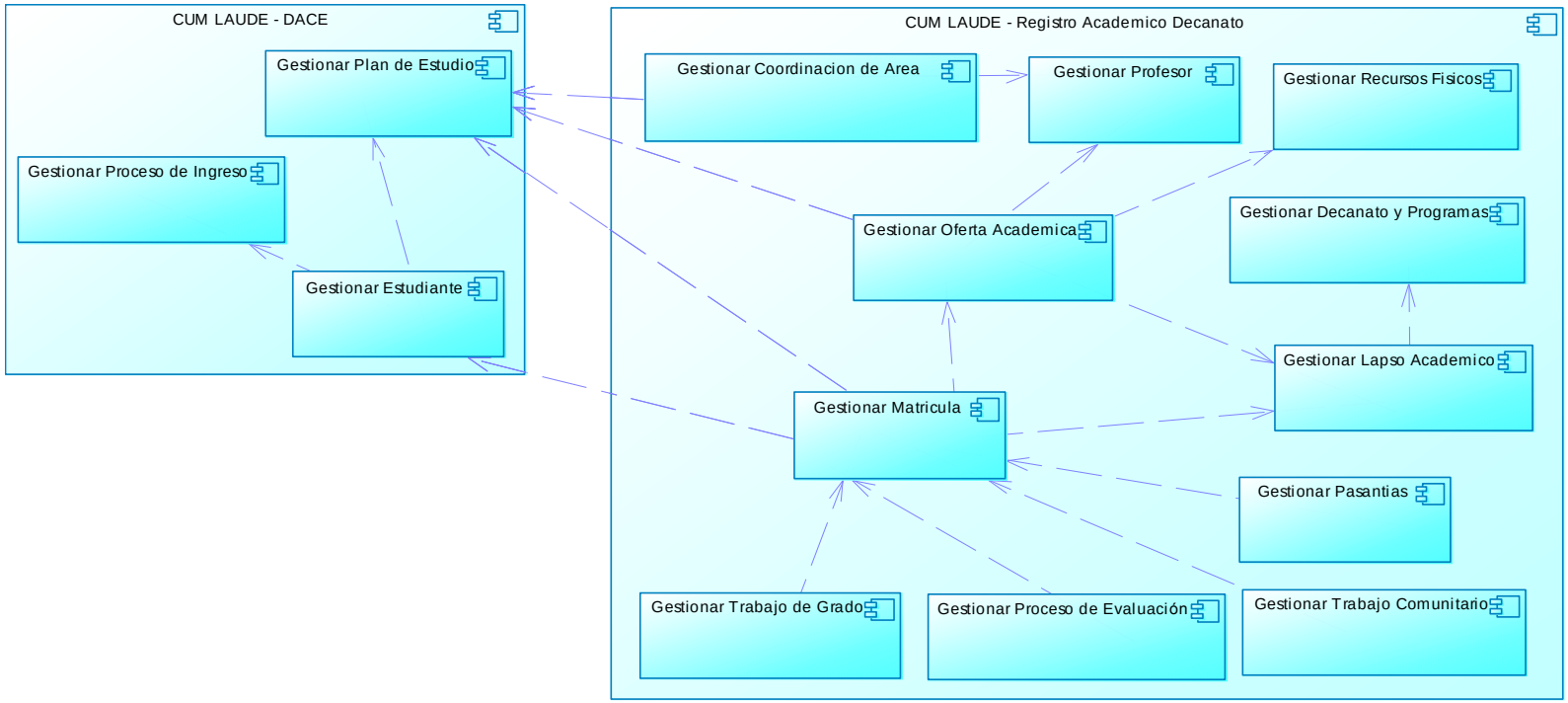

Figura 15. Diagrama de Componentes Detallado - Sistema CUM LAUDE
Fuente: El Autor.

Con la idea de proporcionar una idea de cómo en términos de Clases y Objetos se va a manejar la implementación de cada uno de los componentes de la arquitectura, se hará referencia a lo que propone el Lenguaje de Patrones expuesto en las Bases Teóricas, este además de direccionar la organización y comunicación de las clases de cualquier componente en el sistema, incorpora buenas prácticas aplicadas al desarrollo de aplicaciones empresariales. La Figura 16 muestra como se sugiere abordar el desarrollo de cada componente en la arquitectura.

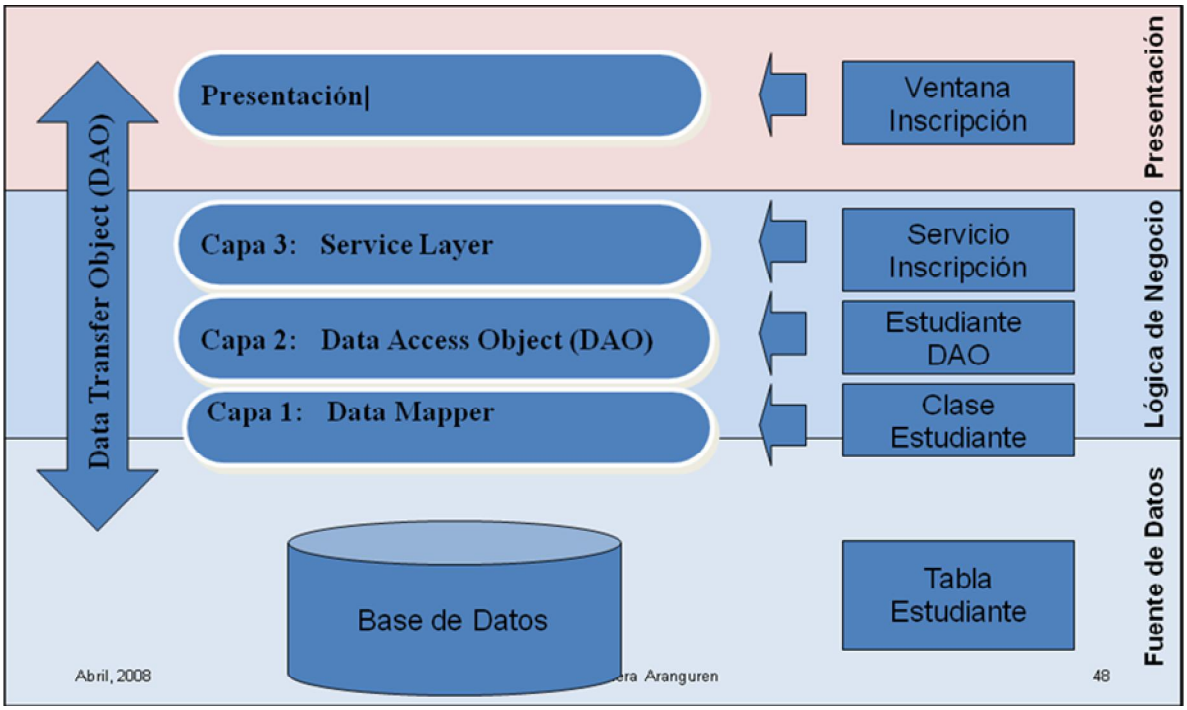

Figura 16. Esquema de Implementación - Componente Sistema CUM LAUDE
Fuente: El Autor.

Como se puede observar la arquitectura desarrollada presenta un disposición por capaz, esto implica que la comunicación entre ellas se hacen al estilo de llamada-retorno, así las capas superiores usan los servicios proporcionadas por las inferiores,

en este caso existe una transferencia de datos en ambos sentidos, es decir entre la capa que solicita y la que procesa, para ello es muy importante estandarizar la comunicación, empleando significados compartidos, allí es donde el modelo de dominio se convierte en una herramienta eficaz, puesto que muestra las entidades que forman el dominio de problema y sus relaciones. En la Figura 17 se puede ver un extracto del modelo de dominio construido para CUM LAUDE.

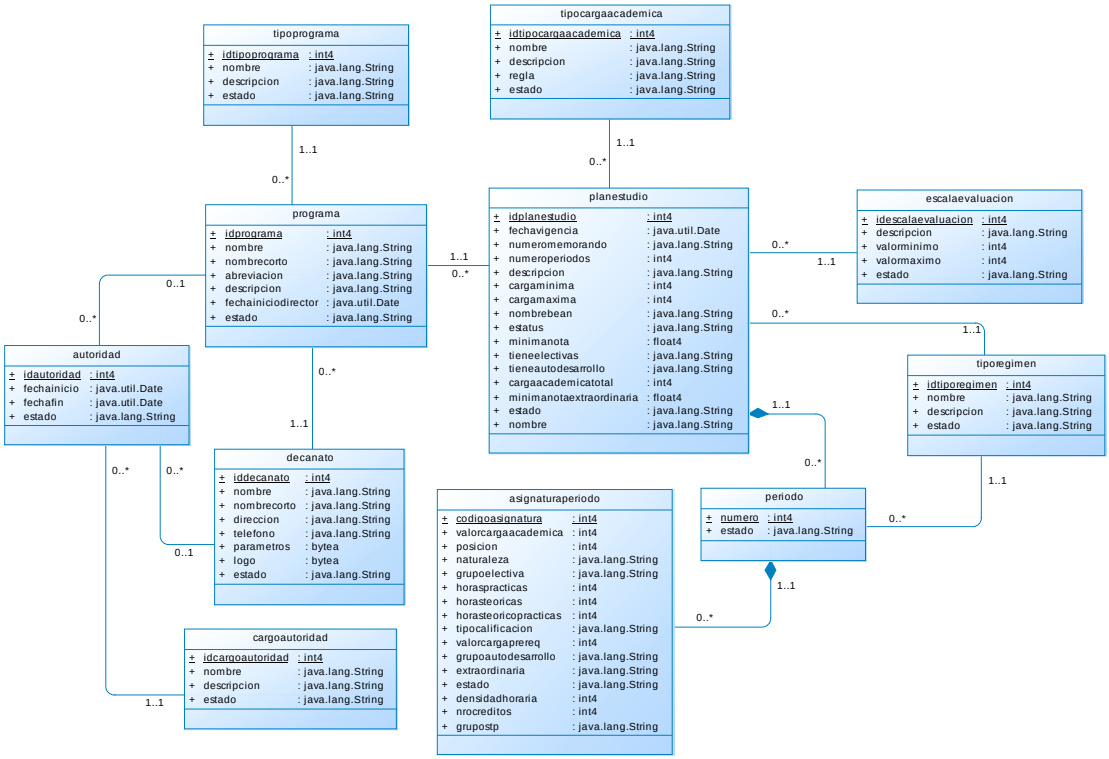

Figura 17. Extracto del Modelo de Dominio - Sistema CUM LAUDE
Fuente: El Autor.

Como se puede ver la lógica de negocio de la aplicación no está acoplada a la presentación, esto trae como beneficio que puede exponerse a otras aplicaciones para que pueda usarse con tecnologías estandarizadas y abiertas como los Web Service, si estas prácticas de diseño se extendieran al desarrollo de aplicaciones de la UCLA se

pudiera generar un ambiente de interoperabilidad, en la búsqueda del reúso de lógicas de negocio disponible en otras aplicaciones, así como también lograr integración de datos, esto puede verse en la Figura 18.

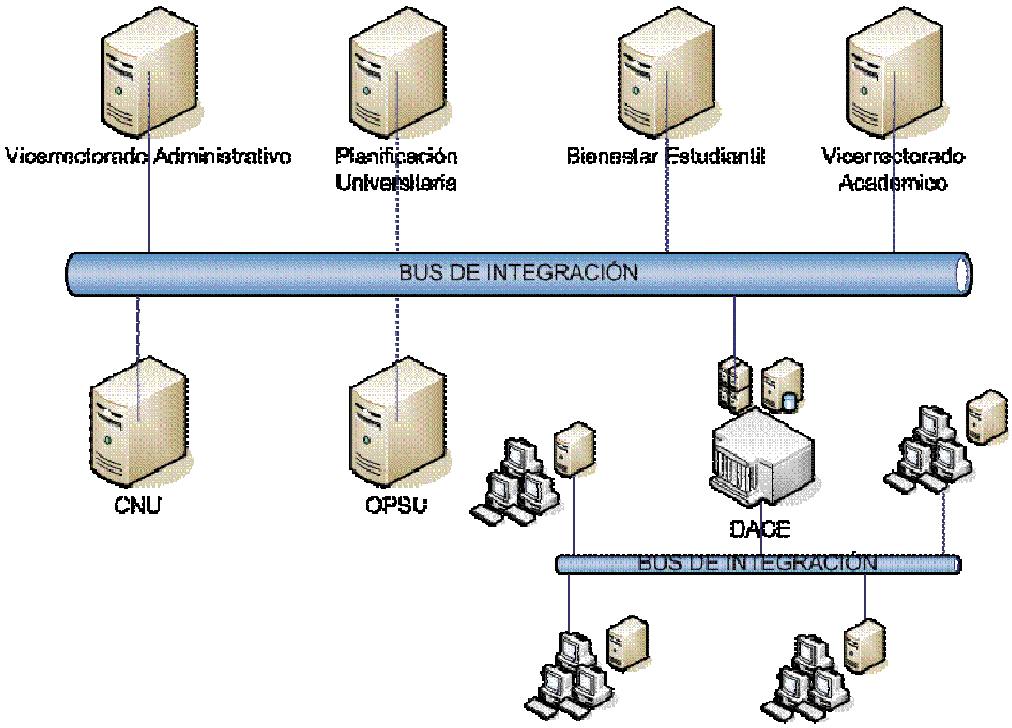

Figura 18. Esquema de Integración en UCLA
Fuente: El Autor.

Se sugiere al momento de desarrollar la aplicación, trabajar en la procura de componentes estandarizados que reúnan el común denominador de las distintas variantes de procesos académicos llevados a cabo en los diversos Decanatos de la UCLA, de tal manera que al llegar el momento de su implementación en la localidad correspondiente pueda adaptarse con facilidad y rapidez. (Ver Figura 19).

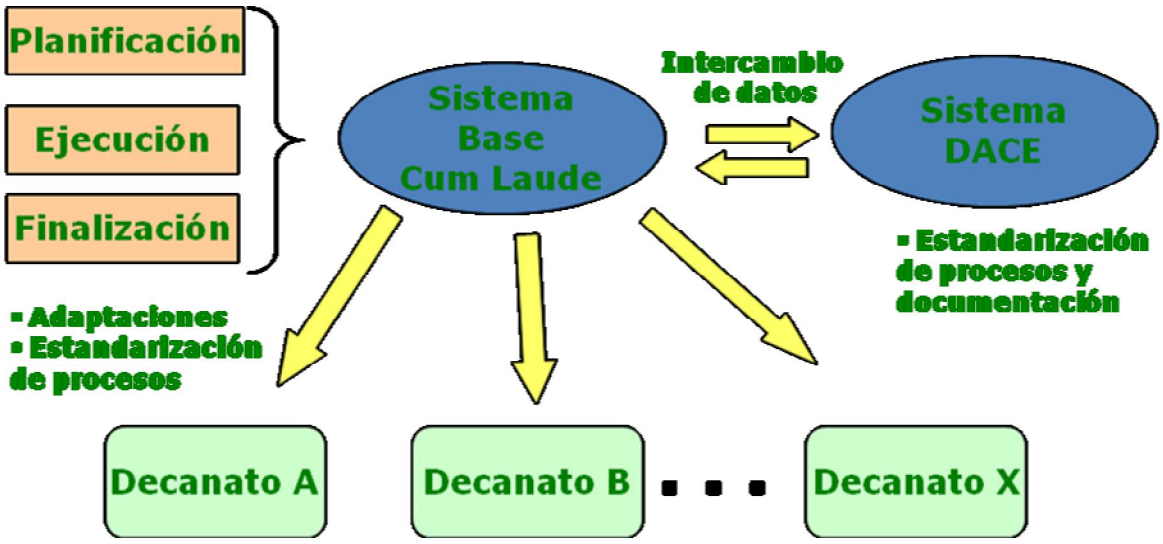

Figura 19. Esquema de Integración en UCLA
Fuente: El Autor.

CONCLUSIONES

Desde el punto de vista funcional CUM LAUDE, presenta una arquitectura que permitirá a cada uno de los decanatos:

- Distribuir automáticamente la información requerida por cada una de los departamentos, unidades o cualquier ente que la requiera en cada Decanato.
- Facilitar mediante el uso de información la toma de decisiones responsable, fundamentada, ágil, acertada y oportuna.
- Proveer información de calidad: precisa, confiable, detallada, oportuna, accesible, integrada y segura.
- Optimización de los procesos y procedimientos de gestión académica y docente a través de la estandarización, agilización, integración y depuración de los mismos.

CUM LAUDE como software de gestión académica único permitirá la estandarización de los procesos y del flujo de información entre los Registros Académicos y la Dirección de Admisión y Control de Estudios para mejorar el servicio prestado a la comunidad universitaria.

Dado que la Dirección de Admisión y Control de Estudios necesita de información acerca del desenvolvimiento de los Registros Académicos, CUM LAUDE establece una plataforma tecnológica que facilitará la obtención de información oportuna, eliminando las barreras de incompatibilidad en el formato y de estructuración de los datos.

Se presento una arquitectura de software que permite un desarrollo de mantenible, soportado por experiencias exitosas de diseño, esto garantiza calidad en el producto final y permite al equipo desarrollador desenvolverse de manera eficaz en un dominio de problema muy dinámico, cifrado por necesidades muy cambiantes.

La implementación de esta arquitectura de solución permitirá sentar las bases para construir dentro de la UCLA software interoperable que rompería con los problemas de consistencia de datos e integración.

El proceso de diseño de la arquitectura se sustento sobre un proceso ágil para generar documentación de valor que pueda ser contundente para resolver los problemas asociados a los procesos de control de estudio y registro académico.

REFERENCIAS BIBLIOGRAFICA

Beck, K. & Fowler, M. (2000). *Planning Extreme Programming.* Addison Wesley.

Yatco, M. (1999). *Joint Application Design/Development*. University of Missouri-St. Louis. [Internet] <http://www.umsl.edu/~sauterv/analysis/JAD.html>

Bass, Clements y Kazman. (1998). *Software Architecture in Practice*. Addison-Wesley.

Clements, P. (1996). *A Survey of Architecture Description Languages*. International Workshop on Software Specification and Design, Alemania.

David Garlan. (2000). *Software Architecture: A Roadmap*. ACM Press.

Garlan, D. y Shaw, M. (1994). *An introduction to software architecture*. CMU Software Engineering Institute Technical Report.

Kruchten, P. (1995). *The 4+1 View Model of Architecture*. IEEE Software.

Garland, J. y Richard, A. (2005). *Large Scale Software Architecture*. Wiley & Sons.

IEEE 1471-2000 (2000) [Internet] <http://www.techstreet.com/cgi-bin/detail?product_id=879737

Zambrano, N & Acosta, A. (2001). *Patrones en Interacción Humano-Computador: Fundamentos Teóricos*. Caracas. Venezuela. Universidad Central de Venezuela. Facultad de Ciencias. Escuela de Computación.

Buschmann, F. (1996). *Pattern Oriented Software Architecture, Volume 1: A System of Patterns*. Willey & Sons.

Christopher ,A. (1977). *A pattern language*. Oxford University Press.

Fowler, M. (2003). *Pattern of Enterprise Application Architecture*. Addison Wesley. 2003