



Universidad Nacional Mayor de San Marcos

Universidad del Perú. Decana de América

Facultad de Ingeniería de Sistemas e Informática
Escuela Académico Profesional de Ingeniería de Sistemas

**Diseño de una arquitectura de software para la
integración de sistemas usando SOA y Web Services en
una institución del rubro educación**

TESINA

Para optar el Título Profesional de Ingeniero de Sistemas

AUTOR

Rocío ROMAINVILLE YANCÁN

ASESOR

Víctor Hugo BUSTAMANTE OLIVERA

Lima, Perú

2008



Reconocimiento - No Comercial - Compartir Igual - Sin restricciones adicionales

<https://creativecommons.org/licenses/by-nc-sa/4.0/>

Usted puede distribuir, remezclar, retocar, y crear a partir del documento original de modo no comercial, siempre y cuando se dé crédito al autor del documento y se licencien las nuevas creaciones bajo las mismas condiciones. No se permite aplicar términos legales o medidas tecnológicas que restrinjan legalmente a otros a hacer cualquier cosa que permita esta licencia.

Referencia bibliográfica

Romainville, R. (2008). *Diseño de una arquitectura de software para la integración de sistemas usando SOA y Web Services en una institución del rubro educación*. Tesina para optar el título profesional de Ingeniero de Sistemas. Escuela Académico Profesional de Ingeniería de Sistemas, Facultad de Ingeniería de Sistemas e Informática, Universidad Nacional Mayor de San Marcos, Lima, Perú.

Este trabajo esta dedicado a toda mi familia
en especial a mis padres.

AGRADECIMIENTOS

Al profesor Víctor Bustamante Olivera, por su orientación y dedicación para que este trabajo cumpla con los objetivos trazados.

A mis colegas y amigos de la Universidad Nacional Mayor de San Marcos, Lizeth Flores y Nina Lope por sus observaciones y porque en todo momento me incentivaron para que culmine este trabajo.

A todas aquellas personas que indirectamente me ayudaron para culminar este trabajo y que muchas veces constituyen un invalorable apoyo.

Y por encima de todo doy gracias a Dios.

Diseño de una Arquitectura de Software para la Integración de Sistemas usando SOA y Web Services en una institución del Rubro Educación

RESUMEN

La presente tesina tiene como objetivo fundamental resolver el problema de la integración de sistemas dentro de una empresa. Esto se debe a que muchas veces la TI no es tomada en cuenta con un propósito estratégico sino solamente operativo, por eso que en una organización encontramos islas de información que satisfacen solo las necesidades de un área específica, y no de toda la empresa. Además el crecimiento vertiginoso de la TI ha creado ambientes heterogéneos, lo que ocasiona una barrera cuando queremos integrar e intercambiar información entre nuestras aplicaciones.

Ante esta problemática se ha diseñado una arquitectura de software que permite la integración de los sistemas en la organización de estudio, basándonos en una Arquitectura Orientada a Servicios y Web Services, como una alternativa de solución a los problemas de integración no solo de las aplicaciones internas de la organización, sino que también nos posibilita la integración con aplicaciones externas que pueden formar parte de nuestra cadena de valor.

Palabras Claves: Integración, Arquitectura de Software, Arquitectura Orientada a Servicios, Web Services.

Design of a Software Architecture for System Integration using SOA and Web Services in an Institution of Education Sector

ABSTRACT

This dissertation is aimed at solving the fundamental problem of integrating systems within an enterprise. It's because of many times IT is not taken into account with a strategic purpose but only operational, that is because of an organization are islands of information that only satisfy the needs of a specific area, and not the entire enterprise. Besides the rapid improve of IT has created heterogeneous environments, causing a barrier when we want to integrate and exchange information between our applications.

Faced with this growing problem has been designed a software architecture that enables the integration of systems in the organization of the study, based on a Service-Oriented Architecture and Web Services, as an alternative solution to the problems of integration not only inside to the organization, otherwise it enables integration with external applications that can be part of our value chain.

Key words: Integration, Software Architecture, Service-Oriented Architecture, Web Services

ÍNDICE

Lista de Figuras	vi
Lista de Tablas	vii

DISEÑO DE UNA ARQUITECTURA DE SOFTWARE PARA LA INTEGRACIÓN DE SISTEMAS USANDO SOA Y WEB SERVICES EN UNA INSTITUCIÓN DEL RUBRO EDUCACIÓN.....	V
---	----------

DESIGN OF A SOFTWARE ARCHITECTURE FOR SYSTEM INTEGRATION USING SOA AND WEB SERVICES IN AN INSTITUTION OF EDUCATION SECTOR	VI
--	-----------

CAPÍTULO 1: INTRODUCCIÓN	1
---------------------------------------	----------

1.1	Antecedentes	1
1.2	Definición del Problema	2
1.3	Objetivos	3
1.3.1	Objetivo Principal.....	3
1.3.2	Objetivos Secundarios.....	3
1.4	Justificación	3
1.5	Propuesta	8
1.6	Organización de la Tesina	8

CAPÍTULO 2: MARCO TEÓRICO	9
--	----------

2.1	La Tecnología de Información en las Instituciones Educativas	9
2.1.1	El impacto en la gestión	9
2.1.2	El impacto en la educación.....	10
2.2	Un breve vistazo a la Integración de Sistemas.....	11
2.2.1	Tipos de Soluciones de Integración.....	11
2.2.2	Niveles de Integración.....	13

2.3	La Arquitectura de Software.....	17
2.3.1	Definiciones	17
2.3.2	El Ciclo de vida de la Arquitectura de Software	17
2.3.3	Frameworks y Vistas	19
2.3.4	Patrones de Arquitectura	23
2.3.5	Otros conceptos relacionados con la Arquitectura: Patrones Arquitectónicos, Modelos de Referencia o Frameworks y Arquitecturas de Referencia.....	30
2.4	Arquitectura Orientada a Servicios (SOA) y Web Services	31
2.4.1	¿Qué es un servicio?	32
2.4.2	Los colaboradores de SOA	34
2.4.3	El ciclo de vida de SOA	35
2.4.4	La Arquitectura de Referencia SOA.....	39
2.4.5	Modelo de la Arquitectura Lógica SOA.....	40
2.4.6	Las tecnologías de los Web Services.....	42
2.4.7	Enterprise Service Bus (ESB)	45
2.4.8	Beneficios de una Arquitectura Orientada a Servicios	46
2.5	SOA y Web Services para la Integración	47
2.5.1	La Integración Orientada a Servicios (SOI)	48

CAPÍTULO 3: SOLUCIONES PARA LA INTEGRACIÓN DE SISTEMAS.... 50

3.1	Tecnologías Middleware.....	50
3.1.1	Remote Procedure Call (RPC).....	50
3.1.2	Transaction Processing Monitors (TP Monitors)	51
3.1.3	Database Access Technology (DBAT).....	51
3.1.4	Distributed Object Middleware (DOM)	51
3.1.5	Messaging Oriented Middleware (MOM).....	52
3.2	La aparición de los Web Services como evolución de los Middleware	53
3.3	Comparación de las alternativas de solución al problema dado	54
3.3.1	Características de la solución tradicional cliente/servidor.....	54
3.3.2	Características de la solución con una arquitectura orientada a servicios	54
3.3.3	Parámetros a tomar en cuenta para la elección de la alternativa.....	54
3.4	Descripción de la Metodología a Utilizar para el desarrollo del caso de estudio.....	56

CAPÍTULO 4: DESARROLLO DEL CASO..... 59

4.1	La Asociación Los Profesionales Aéreos – Professional Air	59
------------	--	-----------

4.1.1	Visión	59
4.1.2	Misión	59
4.1.3	Objetivos Estratégicos	59
4.1.4	Estructura Orgánica	60
4.2	Problemática Actual.....	61
4.3	Diseño de la Arquitectura de Software para la Integración de los sistemas	62
4.3.1	Procesos de negocio	62
4.3.2	Análisis de Requerimientos	64
4.3.3	Patrones y Framework de Arquitectura a utilizarse	65
4.3.4	Uso de estándares y tecnología.....	69
4.3.5	Vista de Casos de Uso	70
4.3.6	Vista Lógica	77
4.3.7	Vista de Procesos	104
4.3.8	Vista de Implementación	105
4.3.9	Vista de Despliegue	107
4.3.10	Atributos de calidad	107
CAPÍTULO 5: CONCLUSIONES Y TRABAJOS FUTUROS		109
5.1	Conclusiones	109
5.2	Futuros Trabajos.....	109
ANEXOS		113
Especificación de los casos de uso		113

Índice de figuras

1.1	Prioridades de los CIOs	4
1.2	Principales Objetivos para la Integración de Aplicaciones	5
2.1	Esfuerzos de integración dependiendo de la plataforma	12
2.2	Integración cuando se añade un nuevo proceso de negocio en un ambiente heterogéneo	13
2.3	Integración a nivel de datos donde la aplicación A tiene acceso directo a la base de datos de la aplicación B	14
2.4	Integración a nivel de aplicaciones	15
2.5	Arquitectura básica de un EAI	16
2.6	Integración Orientada a Servicios	16
2.7	El framework de Zackman	21
2.8	Arquitectura empresarial TOGAF	21
2.9	El modelo de vistas “4+1” de Kruchten	22
2.10	Patrón cliente servidor N-capas para una aplicación web	24
2.11	Anatomía del patrón de mensajería	25
2.12	Elementos del patrón Broker	26
2.13	Componentes del patrón coordinador de procesos	28
2.14	Relación entre modelos de referencia, patrones arquitectónicos, arquitectura de referencia y arquitectura de software	30
2.15	Los colaboradores de una arquitectura orientada a servicios	34
2.16	El ciclo de vida de SOA	35
2.17	Una vista de la solución de la Arquitectura SOA	39
2.18	Arquitectura Lógica de SOA	40
2.19	Un mensaje SOAP	43

2.20 Descripciones de servicios centralizados en un registro UDDI	44
2.21 Enterprise Service Bus	45
3.1 Evolución de los Web Services	53
4.1 Organigrama de Professional Air	60
4.2 Proceso de Matrícula	62
4.3 Proceso de Pago de Planillas	63
4.4 Descripción de la Funcionalidad del BizTalk Server	68
4.5 Diagrama de casos de uso	70
4.6 Casos de uso del módulo de gestión de usuarios	71
4.7 Casos de uso del módulo de gestión de personal	72
4.8 Casos de uso del módulo de gestión de alumnos	73
4.9 Casos de uso del módulo de consulta de horarios de docente	73
4.10 Casos de uso del módulo de generación de asientos contables	74
4.11 Casos de uso del módulo de gestión de carreras	75
4.12 Casos de uso del módulo de consolidación de asistencia de personal y docentes	76
4.13 Casos de uso del módulo de gestión de modalidades de matrícula	76
4.14 Diagrama de paquetes del módulo de gestión de usuarios	77
4.15 Clases interfaz del módulo de gestión de usuarios	77
4.16 Definición del servicio a exponer de gestión de usuarios	78
4.17 Diagrama de Clases de negocio del módulo de gestión de usuarios	78
4.18 Diagrama de Clases entidad del módulo de gestión de usuarios	79
4.19 Diagrama de paquetes del módulo de gestión de personal	79
4.20 Diagrama de Clases interfaz del módulo de gestión de personal	80
4.21 Definición del servicio de gestión de personal	80
4.22 Diagrama de Clases de negocio del módulo de gestión de personal	81
4.23 Diagrama de Clases entidad del módulo de gestión de personal	82

4.24 Principales diagramas de secuencia del módulo de gestión de personal	83
4.25 Principales diagramas de secuencia del módulo de gestión de personal	84
4.26 Diagrama de secuencia del caso de uso Listar Tipo Planillas	84
4.27 Diagrama de paquetes del módulo de gestión de alumnos	85
4.28 Definición del servicio a exponer del módulo de gestión de alumno	85
4.29 Diagrama de Clases de negocio del módulo de gestión de alumnos	86
4.30 Diagrama de Clases entidad del módulo de gestión de alumnos	87
4.31 Diagrama de secuencia del caso de uso Crear Alumno	88
4.32 Diagrama de paquetes del módulo de consulta de horarios de docentes	89
4.33 Definición del servicio a exponer del módulo de consulta de horarios de docentes	89
4.34 Diagrama de Clases de negocio del módulo de consulta de horarios de docentes	89
4.35 Diagrama de Clases entidad del módulo de consulta de horarios de docentes	90
4.36 Diagrama de secuencia del caso de uso Consultar Horario Docente	90
4.37 Diagrama de paquetes del módulo de generación de asientos contables	91
4.38 Definición del servicio a exponer del módulo de generación de asientos contables	91
4.39 Diagrama de Clases de negocio del módulo de generación de asiento contable	92
4.40 Diagrama de Clases entidad del módulo de generación de asiento contable	93
4.41 Diagrama de secuencia de Generar asiento contable	94
4.42 Diagrama de paquetes del módulo de gestión de carreras	95
4.43 Diagrama de Clases interfaz del módulo de gestión de carreras	95
4.44 Definición del servicio a exponer del módulo de gestión de carreras	95

4.45 Diagrama de Clases de la capa de negocios del módulo de gestión de carreras	96
4.46 Diagrama de Clases entidad del módulo de gestión de carreras	96
4.47 Diagrama de secuencia del caso de uso consultar carrera.	97
4.48 Diagrama de paquetes del módulo de consolidación de asistencia de personal y docente	98
4.49 Definición del servicio a exponer del módulo de consolidación de asistencia de personal y docente	98
4.50 Diagrama de Clases de la capa de negocios del módulo de consolidación de asistencia de personal y docente	99
4.51 Diagrama de Clases entidad del módulo de consolidación de asistencia de personal y docentes	100
4.52 Diagrama de secuencia del caso de uso consolidar asistencia	101
4.53 Diagrama de paquetes del módulo de gestión de modalidades	102
4.54 Diagrama de Clases interfaz del módulo de gestión de modalidades	102
4.55: Definición del servicio a exponer del módulo de gestión de modalidades	102
4.56 Diagrama de Clases de la capa de negocio del módulo de gestión de modalidades	103
4.57 Diagrama de Clases entidad del módulo de gestión de modalidades	103
4.58 Diagrama de secuencia principales del modulo de gestión de modalidades	104
4.59 Diagrama de la arquitectura de procesos para la solución planteada	105
4.60 Diagrama de componentes para la integración de los sistemas	106
4.61 Vista de despliegue de la solución de integración	107

Índice de tablas

1.1	Mejoras ocasionadas por la integración de sistemas.	6
1.2	Casos de éxito usando SOA	7
2.1	Las vistas en los frameworks	20
2.2	Atributos de calidad para el patrón Cliente Servidor N-Capas	25
2.3	Atributos de calidad para el patrón de mensajería	26
2.4	Atributos de calidad para el patrón broker	28
2.5	Atributos de calidad para el patrón coordinador de procesos	29
3.1	Comparación de las soluciones de acuerdo a los parámetros	56
3.2	Resumen de la metodología a utilizar	58
4.1	Sistemas de información en Professional Air	61
4.2	Listado de servicios a exponer	67

Capítulo 1: Introducción

1.1 Antecedentes

Las instituciones educativas de formación superior tienen una labor muy importante en las sociedades, ya que son la fuente principal de generación de conocimientos e investigación, que son muy relevantes para el desarrollo de un país.

Es por eso muy importante tener en cuenta que la calidad e innovación de los servicios que las instituciones educativas ofrecen debe ir creciendo ascendentemente para ser más competitivas en el mercado y garantizar su permanencia.

Para ofrecer estos servicios de calidad, también hay que tomar en cuenta la adecuada organización y administración de sus procesos internos. Al igual que el resto de empresas, éstas también se acogieron a los cambios operativos que ofrecían las tecnologías de información. La introducción de la computadora y los sistemas de información permitieron a las instituciones educativas automatizar los procesos rutinarios de los diferentes departamentos, reducción de costos de personal, la minimización de errores y la maximización de la velocidad con la que se manejan los datos.

Además el nacimiento de Internet ha impactado de tal manera a los modelos educativos que aún hoy se siguen generando cambios. Ahora los estudiantes tienen mayor acceso a fuentes de información, los docentes están utilizando la TI como herramienta para la enseñanza, y se están abriendo más oportunidades de innovación de servicios debido a los nuevos canales de comunicación generados por la red (email, foros, chats, intranets, etc).

Sin embargo muchas veces el empleo de la TI no es estructurado ni planificado. Esto ocasiona que en muchas instituciones educativas no conozcan el valor real y el potencial de las tecnologías de información, para la calidad de sus procesos o servicios. La TI no sólo se utiliza actualmente para aumentar la productividad, sino que también sirve como propósito estratégico en la gestión de información, y al no ser concebida como tal, es donde se originan los problemas de integración ya que los sistemas se desarrollan en formas aisladas y si están aislados la efectividad de la información para la toma de decisiones es ineficiente. Además que el vertiginoso avance de la tecnología ha

ocasionado que estos sistemas estén desarrollados en ambientes heterogéneos, ocasionando así una mayor problemática para la comunicación entre ellos.

La necesidad de la integración de sistemas dentro de una institución educativa, surge a medida que los directivos van tomando conciencia del verdadero papel de la TI dentro de ellas, no sólo como herramienta de apoyo operacional, sino también una herramienta estratégica para generar valor agregado y competitividad.

1.2 Definición del Problema

Professional Air es una institución que cuenta con 10 años en el mercado y se dedica a la formación de profesionales en Aviación Civil. Cuenta con varios sistemas de información que le permiten soportar su carga operativa, pero éstos no se encuentran integrados, es decir, no hay intercambio de información entre ellos y por lo tanto no satisfacen todos los requerimientos del negocio. Esto se debe a que no existió un planeamiento para el desarrollo de las soluciones, sólo se buscó satisfacer las necesidades específicas de cada departamento.

Cada uno de estos sistemas está aislado de los otros, esto ocasiona que los datos muchas veces tienen que ser ingresados en varios sistemas de diferentes áreas. Un ejemplo particular podría ser al matricularse un alumno debe ser ingresado nuevamente en el sistema de tesorería y en el de contabilidad. Lo mismo ocurre cada vez que un nuevo profesor ingresa a trabajar; éste debe ser registrado en el sistema de tesorería, en el sistema académico y en el sistema de control de asistencia.

El hecho de no haber una comunicación entre los sistemas ocasiona que se generen incoherencias. Por ejemplo cuando un alumno o docente se retira de la institución, debe de ser eliminado o desactivado de todos los sistemas en el que aparece, pero como se hace de manera manual en cada uno de ellos, la falta de comunicación entre las áreas ocasiona que se haga en unos sistemas pero en otros no. Otro ejemplo podría ser cuando un alumno pide un permiso al área académica por ausencia de varios meses por viaje, salud, etc. El Departamento Académico le aprueba la solicitud, se ingresa en el sistema académico pero no se informa a Tesorería, entonces el alumno al retornar sus clases aparece como deudor en el sistema de Tesorería y está sujeto a moras e intereses.

Es así que se generan muchos errores y problemas entre las áreas que no comparten el flujo de información ya que no sólo se afectan ellas mismas sino también a los alumnos que son los principales clientes.

1.3 Objetivos

1.3.1 Objetivo Principal

El objetivo principal de esta tesina es el diseño de una arquitectura para resolver el problema de integración en la institución educativa “Professional Air” usando SOA y Web Services.

1.3.2 Objetivos Secundarios

- Definir los procesos de negocio para los cuales se va a realizar la integración.
- A partir de la definición de los procesos identificar los requerimientos funcionales.
- Identificar los requerimientos no funcionales para el proyecto de integración.
- Identificar de cada sistema que servicios se necesitan publicar para habilitar la integración.
- Seleccionar patrones de diseño y el framework a usarse en la definición de la arquitectura.
- Aplicar los conceptos de SOA para definir una arquitectura de integración.

1.4 Justificación

Hoy en día es frecuente encontrar instituciones que cuentan con sistemas de información que han sido desarrollados para satisfacer necesidades específicas, muchas veces usando plataformas, lenguajes y bases de datos diferentes y que no se encuentran comunicados. Las aplicaciones se encuentran aisladas y por el hecho que son especializadas se encuentran limitadas y no cumplen con los objetivos y estrategias del negocio. Por eso una las de las prioridades de los CIOs, como se muestra en la figura 1.1, es la integración.

Además los ambientes tecnológicos han evolucionado muy rápidamente, esto ha generado que de alguna forma los sistemas en una empresa se hayan desarrollado en

ambientes heterogéneos y esto ha producido cierto nivel de complejidad para su integración.

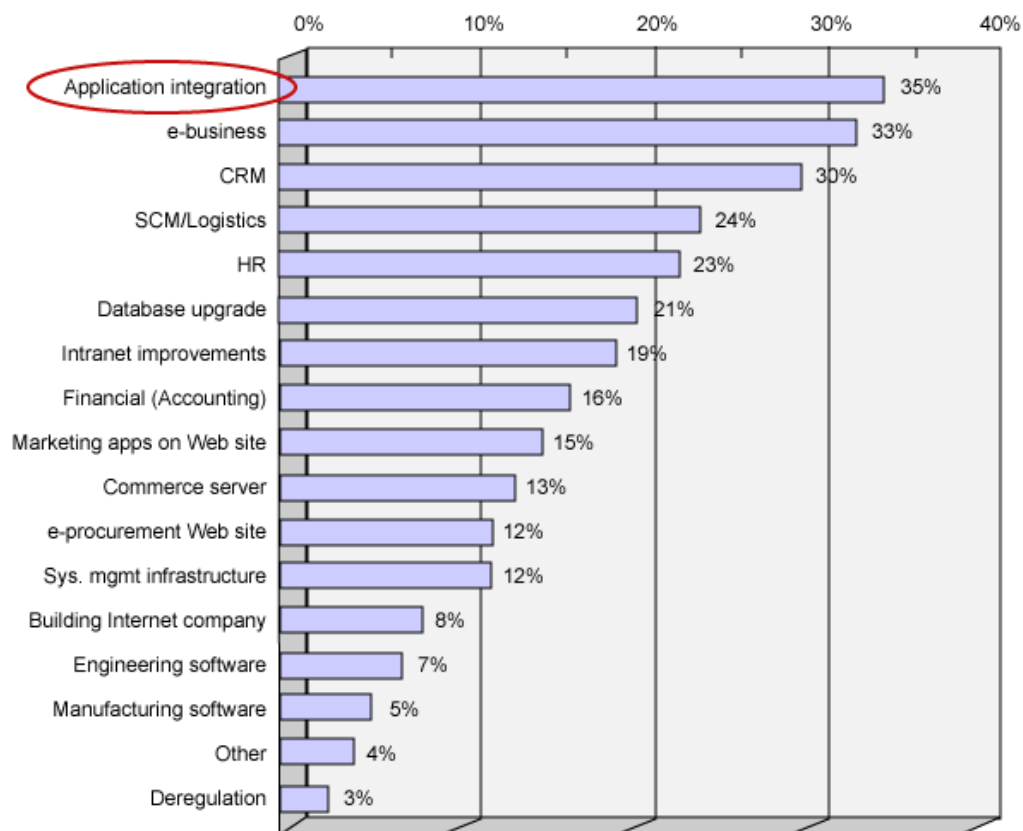


Figura 1.1: Prioridades de los CIOs [Holley+03]

Trabajar en un ambiente integrado va a permitir a la empresa ir en una sola dirección, ser más eficientes, tomar las decisiones correctas y también disminuir los recursos.

Como institución educativa, debe ser prioridad ofrecer un servicio de calidad al alumnado. Por lo tanto, si no existe una integración los esfuerzos se van a concentrar aún en temas operacionales en lugar de mejorar la calidad de los servicios de educativos. Así podemos observar en la figura 1.2 que el principal objetivo de la integración de aplicaciones en las organizaciones es la eficiencia operacional.

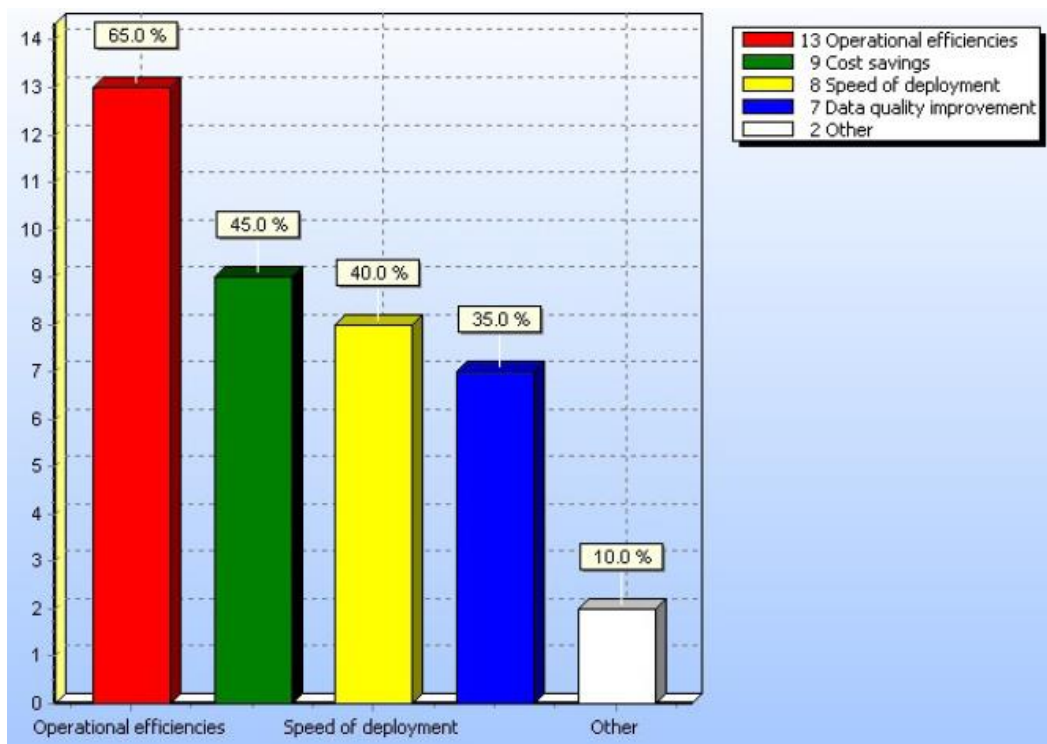


Figura 1.2: Principales objetivos para la integración de aplicaciones [Zhai03]

Actualmente Professional Air cuenta con varios sistemas de soporte a la gestión educativa y administrativa pero han sido desarrollados en manera aislada. Esto ocasiona una serie de problemas que se solucionarían o mejorarían al contar con un ambiente integrado como se muestra en la siguiente tabla:

Mejora	Descripción
Distribución de la información	Al tener los sistemas integrados, la información va a poder ser distribuida a lo largo de toda la institución educativa, permitiendo a los diferentes departamentos compartir la información, por ejemplo, el departamento académico debe estar informado de los pagos de los alumnos, para poder deducir las causas de inasistencias.
Agilidad para la toma de decisiones	Al tener un sistema integrado, se tendrá mayor visión y disponibilidad de la información en el momento que se requiera.
Disminución de tiempos	La información tiene que ser ingresada en varios sistemas (por ejemplo los datos del alumno son ingresados en el sistema de matrícula y también en el sistema de tesorería) o tiene que ser consultada en un sistema para pasarse a otro (por ejemplo la asistencia de los docentes en el sistema de control de asistencia para pasarse al sistema de planillas). Con la integración se solucionarían estos problemas.
Rápida integración de nuevos sistemas	Al establecer una arquitectura para la integración con SOA, es mucho más sencilla la incorporación de nuevos sistemas, ya que pueden reutilizarse los servicios que están disponibles.

Tabla 1.1: Mejoras ocasionadas por la integración de sistemas

Al escoger SOA como herramienta de solución además de habilitar la integración, va a proporcionar, debido a sus características, un ambiente flexible y que se acomode a los cambios. Es así que la institución educativa va a estar preparada para evolucionar de acuerdo a las exigencias del mercado.

A continuación vamos a presentar unos casos de éxito usando SOA como herramienta para la integración de instituciones en general, pero que nos van a dar un alcance del tipo de solución y los beneficios que se logran:

Empresa	Descripción de la solución SOA
Empresa: Banco Hipotecario País: Argentina	El negocio principal del banco eran los créditos hipotecarios, pero debido a boom que tuvieron en ventas de tarjetas de crédito, de cuentas corrientes y demás quisieron enfocarse también en otro tipo de negocio. Se inició la implementación de 4 nuevos canales de atención al cliente y la arquitectura de integración utilizada fue SOA. El beneficio principal de contar con una arquitectura de estas características está dado por la reutilización de los servicios. Por ejemplo, si esta infraestructura tiene un servicio de consulta de saldo, cualquier aplicación o cualquier canal bancario que necesita consumir este servicio, en lugar de hacer un pedido a la aplicación que lo genera, lo pide directamente a esta arquitectura, y ésta es la que se encarga de entregarle ese dato. Esto tiene una serie de ventajas importantes. En primer término, es más sencillo, porque se puede reutilizar ese servicio tantas veces como se quiera. Al mismo tiempo se reducen los tiempos para desarrollar software, porque determinadas cosas ya vienen dadas por estos servicios. El gerente sostuvo que, gracias a la utilización de esta tecnología, el banco en sólo 4 meses pudo subir en producción. A los 120 días ya tenían funcionando el call center, a los 150 días las terminales de autoservicio, a los 180 el IVR y a los 210 el home banking. Esto es un tiempo récord en el mundo de la tecnología de implementación de canales. Si el banco no hubiera utilizado una arquitectura SOA, habría tenido que desarrollar interfases entre todas las aplicaciones, por lo que tendría que haber trabajado no menos de 4 veces más porque, al haber implementado 4 canales, tendría que haber desarrollado en forma artesanal tantas interfases como canales tenía que conectar. En cambio, con la arquitectura SOA, lo hizo una sola vez y lo reutilizó para toda la organización.
Empresa: Grupo Electra País: México	Al ser una cadena que opera 1,800 tiendas de retail en México tenían la necesidad de utilizar tecnologías que permitieran la comunicación entre las tiendas, así como la conexión de información a los diferentes proveedores tanto internos como externos que utilizan multiplataformas. Con la implementación de SOA, el Grupo Elektra resolvió sus problemas de comunicación, ahora cuentan con una estructura que les permite que todas sus tiendas se comuniquen a la infraestructura central del site. Este sistema hace las peticiones tanto de sus proveedores internos como externos. Los beneficios que les trajo esta implementación fueron que ahora existe una mayor posibilidad de hacer cambios, respetando la interfase de los componentes de los servicios que les brindan y vía XML. SOA ofrece un buen nivel de flexibilidad, y hace posible la interoperabilidad entre negocios.
Empresa: State University of New York País: Estados Unidos	La Universidad de New York, que engloba 64 centros, 30000 miembros de facultades y 414000 estudiantes, integró sus registros dispares y sus sistemas de e-learning en una ambiente común usando SOA. Anteriormente el programa de enseñanza virtual de la universidad fue construido sobre un sistema centralizado en Lotus Notes/Domino. A medida que el sistema fue creciendo la integración del sistema de educación a distancia con otros sistemas del campus demostró ser un verdadero reto, sobre todo con la identidad del estudiante. Cada escuela tiene su propio sistema de información estudiantil, y los estudiantes pueden tomar cursos presenciales y en línea lo que llevaba a un doble proceso de registro, resultando en una doble matrícula.

Tabla 1.2: Casos de éxito usando SOA

Las instituciones educativas deben valorar la integración de sistemas, ya que es una necesidad y una obligación si queremos generar ventajas competitivas tanto a nivel operacional como estratégico.

1.5 Propuesta

Ante el problema planteado y luego de revisar la bibliografía y los recursos encontrados en Internet, vemos que la solución mas apta para el problema de integración es SOA como arquitectura y Web Services como tecnología para implementar esa arquitectura.

Se escogió SOA debido a los beneficios que nos plantea, ya que está basada en servicios que se pueden coordinar para formar flujos de procesos de negocios. Cada servicio es independiente a otro, lo que permite que la comunicación entre estos sea de acoplamiento débil. Otra cosa importante es que estos servicios pueden ser reusables y orquestarse con otros para formar un componente mayor.

Se escogió el uso de Web Services, ya que operan en base a protocolos y tecnologías estándar. Estos estándares están implementados en las plataformas y productos de los principales proveedores de software.

1.6 Organización de la Tesina

El propósito del presente capítulo es de dar a entender al lector una clara comprensión de cual es el problema que se quiere solucionar, la importancia y el impacto de la solución. En el Capítulo II revisaremos los conceptos necesarios para poder llegar un entendimiento suficiente de los elementos que conforman la solución propuesta. A continuación en el Capítulo III, daremos un vistazo a las diferentes tecnologías utilizadas para la integración de sistemas. En el capítulo IV procederemos al aporte teórico-práctico, seguido del capítulo V donde se mencionarán las conclusiones y los futuros trabajos y por último el capítulo VI de las referencias bibliográficas.

Capítulo 2: Marco Teórico

El propósito del siguiente capítulo es brindar al lector información necesaria para que pueda tener un mejor entendimiento de los conceptos usados en la presente tesina para entender el problema y la solución planteada.

2.1 La Tecnología de Información en las Instituciones Educativas

El impacto de las TI en el sector educativo se manifiesta de una manera muy especial en las actividades laborales y en el mundo educativo, ya que ha habido cambios en la forma de enseñar y de aprender, las infraestructuras y los medios que utilizamos para ello, la estructura organizativa de la institución y su cultura.

2.1.1 El impacto en la gestión

En el ámbito de la gestión, la TI puede automatizar y descentralizar la gestión de la institución educativa de una manera coordinada. La integración de nuevas tecnologías en las instituciones educativas también se manifiesta con su uso intensivo en los trabajos de administración y de gestión (bases de datos centralizadas para evitar la duplicación de trabajos, automatización de tareas, etc), buscando formas de organización y funcionamiento de las instituciones más eficaces, eficientes, ágiles, flexibles y con una mayor descentralización administrativa. Algunas tareas que actualmente se realizan con el apoyo de la TI son:

- Gestión de secretaría académica: matrícula, expedientes de alumnos, certificados, títulos, reportes de notas, relaciones con la administración educativa (nóminas, actas de evaluaciones), documentación administrativa en general.
- Gestión del personal: fichas del personal, control de asistencia.
- Administración general de la institución: contabilidad, tesorería, facturación, previsión de pagos, inventarios, etc.
- Gestión de tutorías: seguimiento y evaluación de los estudiantes.
- Apoyo a la dirección académica: gestión de horarios (cursos, profesores, aulas).
- Gestión de bibliotecas: catálogos, préstamos.

Sin embargo con el surgimiento de Internet y las nuevas formas de comunicación, se generaron nuevas aportaciones a la gestión educativa. Algunas son:

- Mejor coordinación entre los diversos servicios.
- Proporcionar completa información sobre todos los aspectos relacionados con la institución educativa, sus servicios y titulaciones a través de un espacio Web institucional.
- Realización de múltiples trámites administrativos desde Internet: matrículas, consulta de notas, etc.
- Comunicación ágil de la administración con los estudiantes y con el profesorado a través de los oportunos nuevos canales de comunicación. Progresiva sustitución de las comunicaciones en papel.

Podemos observar que el impacto de la TI en las labores de gestión esta cambiando la forma de hacer las cosas, y las instituciones educativas deben asumir esos cambios para ser competitivos en el mercado.

2.1.2 El impacto en la educación

El uso generalizado de la TI conlleva cambios y efectos sobre las actividades laborales y el mundo educativo como por ejemplo la formación básica que precisamos las personas, la forma de enseñar y de aprender, las infraestructuras y los medios que utilizamos para ello.

A continuación vamos a nombrar algunos efectos que ha tenido la TI sobre aspectos relevantes en la educación:

- La importancia creciente de la educación informal. Con el nacimiento de Internet, instituciones como museos, bibliotecas y centros de recursos utilizan cada vez más esta tecnología para difundir material. Los jóvenes cada vez saben más y aprenden más cosas fuera de la escuela. Por ello uno de los retos que tienen las instituciones educativas consiste en integrar las aportaciones de estos canales formativos en los procesos de enseñanza y aprendizaje.
- Nuevos contenidos curriculares. En los planes de estudios se van incorporando la alfabetización digital básica y diversos contenidos relacionados con el aprovechamiento de las TI en cada materia.

- Nuevos entornos virtuales de aprendizaje. Gracias a la TI se pueden ofrecer nuevos entornos para la enseñanza y el aprendizaje libres de las restricciones que imponen el tiempo y el espacio en la enseñanza presencial, y capaces de asegurar una continua comunicación virtual entre estudiantes y profesores.
- Necesidad de una formación didáctico-tecnológica del docente. Los docentes también necesitan una alfabetización digital y una actualización didáctica que le ayude a conocer, dominar e integrar los instrumentos tecnológicos en su práctica docente.

2.2 Un breve vistazo a la Integración de Sistemas

Para entender el concepto de integración de sistemas, primero hay que definir qué es integración. Integración es la conexión y la colaboración mutua entre actividades o procesos, lo que sugiere que la integración es más que una actividad simple de unir partes; la integración debe permitir que esas partes se unan de forma coherente.

Actualmente en las organizaciones, el término integración tiene 2 vertientes que están fuertemente relacionadas: la integración del negocio y la integración de sistemas. La integración de sistemas puede ser una restricción para la integración del negocio. Se define integración del negocio como la creación de actividades de negocio coordinadas, manejadas por diferentes personas, en grupo o por organizaciones de negocio. La integración de sistemas significa que las aplicaciones se combinan o funcionan dentro de una facilidad, en lugar de tener hardware, software y comunicación independientes.

2.2.1 Tipos de Soluciones de Integración

Los procesos de negocio se han vuelto la clave principal de las organizaciones y éstas están demandando que la tecnología vaya de la mano de sus procesos. El nivel de inteligencia automática de los negocios determina el alto nivel de configuración de una solución de integración. La mayoría, según [Erl04], puede ser agrupada en dos categorías:

- Ampliación de una aplicación existente.
- Nuevos procesos de negocio.

2.2.1.1 Ampliación de una aplicación existente

Una aplicación existente es ampliada para soportar nuevas características o la modificación de un proceso de negocio. Esta ampliación requiere que la aplicación tenga acceso a datos externos o interfaces con lógica de negocio que residen en otra aplicación. Para evitar data redundante o volver a desarrollar una funcionalidad de negocio existente, se requiere un canal de integración que conecte a las dos aplicaciones.



Figura 2.1: Esfuerzos de integración dependiendo de la plataforma.

El esfuerzo de integración en este caso depende, como vemos en la Figura 2.1, de la plataforma de las aplicaciones cuando son homogéneas o heterogéneas.

2.2.1.2 Nuevos procesos de negocio

Algunas veces surgen cambios drásticos en los procesos de negocio existentes en una organización. Puede deberse, por ejemplo, a una reestructura organizacional o a la expansión de las áreas de servicio. Esto genera que algunos procesos sean combinados, otros consolidados o son introducidos nuevos procesos.

Estos cambios fundamentales pueden requerir también cambios en los ambientes técnicos responsables de automatizar las áreas de negocio afectadas.

Incorporar nuevos procesos de negocio, introduce una nueva dimensión para la integración de aplicaciones, especialmente cuando el proceso requiere un número de aplicaciones heterogéneas para interactuar como se muestra en la Figura 2.2.

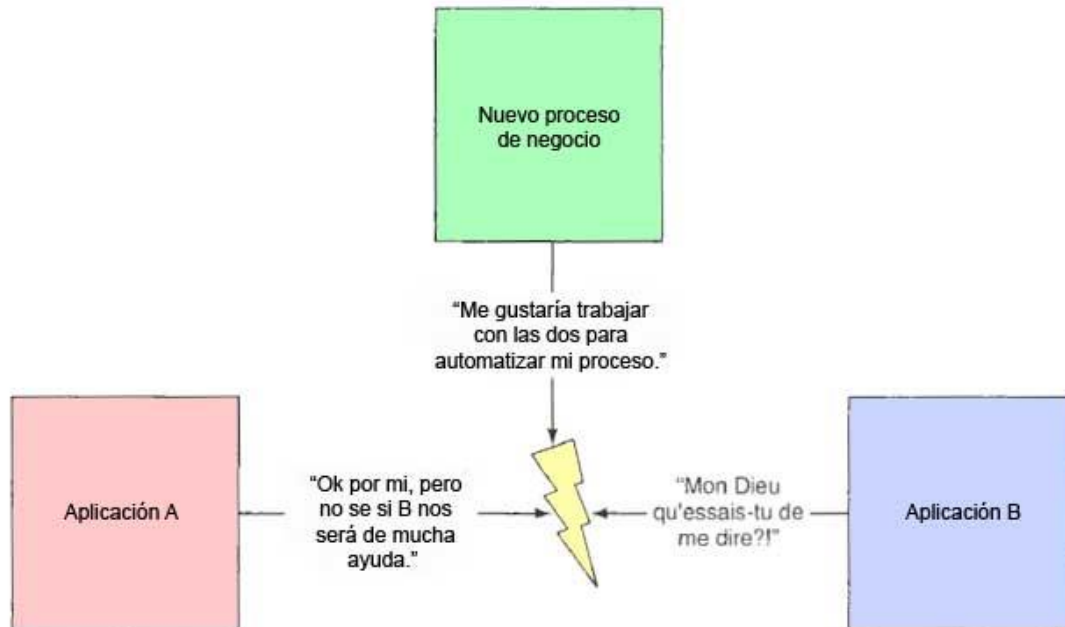


Figura 2.2: Integración cuando se añade un nuevo proceso de negocio en un ambiente heterogéneo.

2.2.2 Niveles de Integración

Los esfuerzos iniciales para lograr la interoperabilidad estuvieron centrados en compartir datos a través de la replicación y acceso directo a los repositorios. Las tendencias entonces cambiaron y se orientaron hacia lo lógica o a la aplicación, que no sólo mejoraba los requerimientos de intercambiar datos sino que también incorporaba las reglas de negocio relacionadas a los datos compartidos. [Erl04] considera los siguientes niveles:

2.2.2.1 Integración a Nivel de Datos

Los datos de una aplicación son compartidos sin involucrar su lógica de aplicación. Por ejemplo en la Figura 2.3 vemos que la aplicación A recibe acceso directo a la base de datos de la aplicación B. La lógica de la aplicación B no está involucrada en la transferencia de sus datos.

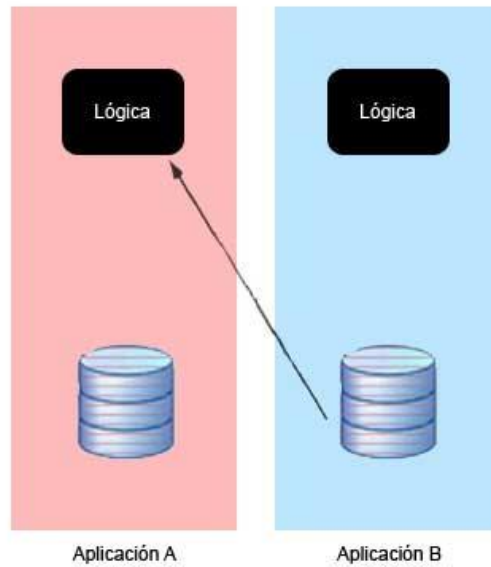


Figura 2.3: Integración a nivel de datos donde la Aplicación A tiene acceso directo a la Base de Datos de la Aplicación B

Otra alternativa de este tipo de integración es donde la base de datos de la aplicación B puede replicarse a la base de datos de la aplicación A.

2.2.2.2 Integración a Nivel de Aplicación

En la figura 2.4 se ve un ejemplo de este tipo de integración. La aplicación A hace una solicitud de información a la aplicación B mediante una interfase directa dentro de la aplicación B. Todas las comunicaciones siguientes son manejadas a través de esta conexión y cualquier lógica de negocio de la aplicación B, es aplicada para el intercambio de datos.

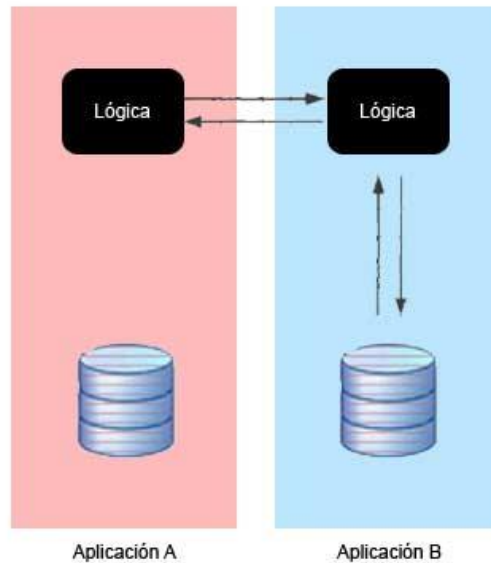


Figura 2.4: Integración a nivel de aplicaciones.

Cuando trabajamos con plataformas de aplicación dispares, algunas veces un paso intermedio es introducido para traducir protocolos de comunicación incompatibles.

2.2.2.3 Integración a Nivel de Procesos

A este nivel, compartir datos o lógica de aplicación facilita un nuevo proceso de negocio automatizado, o la unión de dos o más procesos existentes. Este nuevo proceso es realizado a través de la integración de dos o más aplicaciones.

Tradicionalmente, la integración a nivel de procesos de negocio se ha conseguido a través de una variedad de modelos de integración punto a punto, sin embargo, la llegada de los EAI y de middleware especializados han hecho del modelo de mensajería una opción mas conveniente.

En la figura 2.5 mostramos una arquitectura EAI básica. El broker es utilizado para habilitar la comunicación entre aplicaciones heterogéneas. El componente de orquestación permite la administración y ejecución de nuevos procesos.

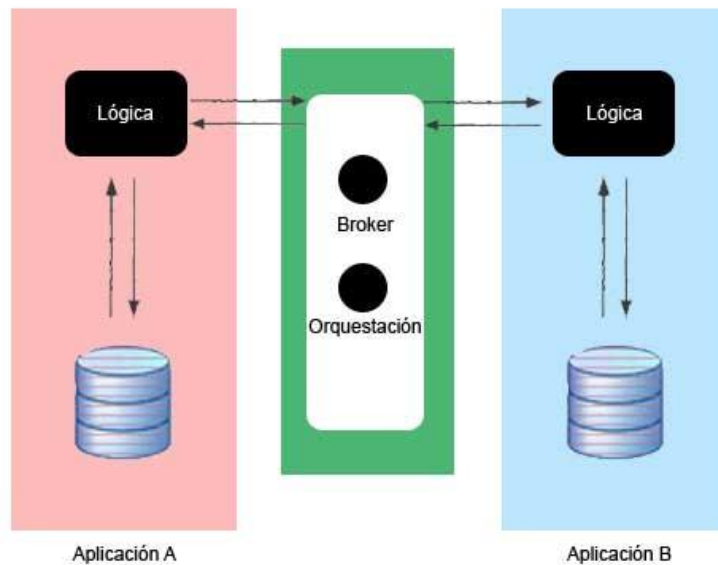


Figura 2.5: Arquitectura Básica de un EAI.

2.2.2.4 Integración Orientada a Servicios

Es importante entender que los Web Services no proporcionan una solución de integración por sí solos. No reemplazan la necesidad de los middleware y muchas tecnologías tradicionales de integración. Los Web Services simplemente agregan nuevos componentes que pueden ser utilizados efectivamente en una variedad de arquitecturas.

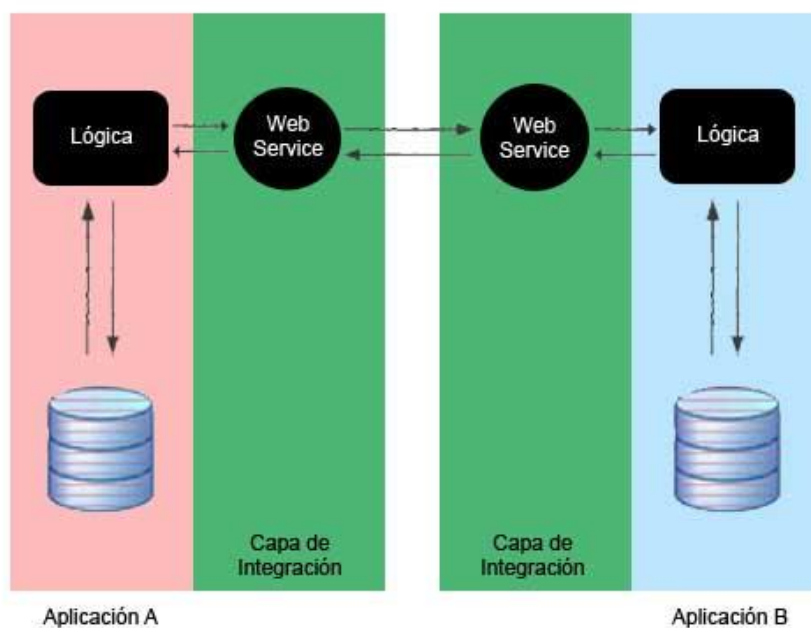


Figura 2.6: Integración orientada a servicios.

La capa lógica de integración, que se muestra en la figura 2.6, creada para exponer APIs legados vía Web Services simplemente ofrece un medio estándar para compartir datos y lógica de programación.

2.3 La Arquitectura de Software

2.3.1 Definiciones

La arquitectura de software esta en crecimiento, pero continúa siendo una disciplina joven; por tanto no tiene una sola definición aceptada. Por eso hemos recopilado algunas de las definiciones encontradas en la bibliografía y que se encuentran también listadas en la Web del SEI (Software Engineering Institute):

- 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. [IEEE00]
- La Arquitectura de Software de un programa o sistema de computación es la estructura o estructuras del sistema, que comprende elementos de software, las propiedades externamente visibles de esos elementos, y la relación entre ellos. [Bass+03]
- La arquitectura de Software va mas allá que los algoritmos y estructuras de datos, diseñando y especificando la estructura global del sistema que emerge como una nueva clase de problema. Los problemas estructurales incluyen organización y estructura de control global; protocolos para la comunicación, sincronización y acceso a datos; asignación de funcionalidad para diseñar elementos; distribución física; composición de elementos de diseño; escalamiento y ejecución; y la selección entre alternativas de diseño. [Garlan+94]

2.3.2 El Ciclo de vida de la Arquitectura de Software

Como indica [Bass+03], las actividades que forman parte del ciclo de vida de la arquitectura de software son las siguientes:

2.3.2.1 Crear los casos de negocio para el sistema

Crear un caso de negocio es mucho mas amplio que simplemente evaluar las necesidades del mercado para un sistema. Es un paso importante en la creación y limitación de cualquier requerimiento futuro.

¿Cuál es el costo del producto? ¿Cuál es el mercado al que se dirige? ¿Cuanto demorará su tiempo de salida al mercado? ¿Necesitará interactuar con otros sistemas? ¿Hay limitaciones en el sistema con las que debemos trabajar? En todas estas preguntas deben participar los arquitectos de sistemas. No debe decidir las solamente el arquitecto, pero si éste no es consultado en la creación de los casos de negocio, podría ser imposible alcanzar los objetivos del negocio.

2.3.2.2 Entender los Requerimientos

Hay una variedad de técnicas para la obtención de los requisitos de los stakeholders. Por ejemplo, el análisis orientado a objetos utiliza escenarios, o “casos de uso” para plasmar los requerimientos. Los sistemas de seguridad críticos usan enfoques más rigurosos, como los modelos de maquinas de estado finitas o lenguajes de especificación formal.

Otra técnica que ayuda a comprender los requerimientos es la creación de prototipos. Los prototipos pueden ayudar a modelar la conducta deseada, el diseño de la interfaz de usuario, o el análisis de la utilización de los recursos. Esto ayuda a hacer el sistema “real” a los ojos de sus stakeholders y pueda rápidamente catalizar las decisiones sobre el diseño del sistema y el diseño de su interfaz de usuario.

Independientemente de la técnica usada para obtener los requerimientos, la calidad deseada del sistema que se construirá determinará la forma de su arquitectura. Tácticas específicas han sido utilizadas por los arquitectos para alcanzar determinados atributos de calidad.

2.3.2.3 Seleccionar una arquitectura

2.3.2.4 Analizar o evaluar una arquitectura

En cualquier proceso de diseño habrá múltiples candidatos. Algunos serán rechazados de inmediato, otros competirán por la primacía. La elección de uno de estos diseños que compiten de forma racional es uno de los mayores retos del arquitecto.

Evaluar una arquitectura por las cualidades que la soportan es esencial para asegurar que el sistema construido sobre esa arquitectura satisfaga las necesidades de los stakeholders. Son cada vez más generalizadas las técnicas de análisis para evaluar los atributos de calidad que imparte una arquitectura a un sistema. Técnicas basadas en escenarios proporcionan uno de los más generales y efectivos enfoques para la evaluación de una arquitectura.

2.3.2.5 Implementar el sistema basado en la arquitectura

Esta etapa tiene que ver con mantener a los desarrolladores fieles a las estructuras y protocolos de interacción limitados por la arquitectura. Tener una arquitectura explícita y bien comunicada es el primer paso para garantizar la conformidad de la arquitectura. Es mejor tener un ambiente o infraestructura que ayude a los desarrolladores activamente en la creación y mantenimiento de la arquitectura.

2.3.2.6 Asegurar que la implementación cumple con la arquitectura

Por último, cuando se crea una arquitectura y se usa, viene una fase de mantenimiento. Se requiere una vigilancia constante para asegurar que la arquitectura real y su representación permanecen fieles una con otra durante esta fase. Aunque el trabajo en ésta área es comparativamente inmadura, ha habido una intensa actividad en los últimos años.

2.3.3 Frameworks y Vistas

Se puede descubrir propiedades diferentes en una arquitectura dependiendo del ángulo con que la mires. Una “vista” representa un aspecto parcial de una arquitectura de software que muestra propiedades específicas de un sistema de software.

Vamos a examinar las propuestas sobre organización de vistas desarrolladas en el contexto de los frameworks más influyentes. Tanto los marcos arquitectónicos como las metodologías de modelado, acostumbran ordenar las diferentes perspectivas de una arquitectura en términos de vistas (views). La mayoría de los frameworks y estrategias reconoce entre tres y seis vistas, que son las que se incluyen en la Tabla 2.1.

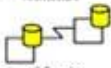
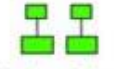
Zachman (niveles)	TOGAF (Arquitecturas)	4+1 (Vistas)
Scope	Negocio	Lógica
Empresa	Datos	Proceso
Sistema Lógico	Aplicación	Física
Tecnología	Tecnología	Desarrollo
Representación		Casos de Uso
Funcionamiento		

Tabla 2.1: Las vistas en los frameworks[Billy04]

2.3.3.1 El framework de Zackman

El marco de referencia para la arquitectura empresarial de John Zachman, fue originalmente concebida en IBM en el año 1980. Identifica 36 vistas de arquitectura (“celdas”) basadas en 6 niveles (scope, empresa, sistema lógico, tecnología, representación detallada y funcionamiento empresarial) y seis aspectos (datos, función, red, gente, tiempo, motivación) (ver Figura 2.7). Las columnas del framework, que no tienen orden de importancia, representan abstracciones únicas de la empresa para reducir la complejidad de cualquier modelo construido.

A FRAMEWORK FOR ENTERPRISE ARCHITECTURE TM

	DATA <i>It's</i>	FUNCTION <i>How</i>	NETWORK <i>Where</i>	PEOPLE <i>Who</i>	TIME <i>When</i>	MOTIVATION <i>Why</i>	
SCOPE (CONTEXT)	List of Things Important to the Business 	List of Processes the Business Performs 	List of Locations in which the Business Operates 	List of Organizations Important to the Business 	List of Events/Cycles Significant to the Business 	List of Business Goals/Strategies 	SCOPE (CONTEXT)
Planner	Entity = Class of Business Thing	Process = Class of Business Process	Node = Major Business Location	People = Major Organization Unit	Time = Major Business Event/Cycle	Ends/Mean = Major Business Goal/Strategy	Strategist
BUSINESS MODEL (CONCEPTS)	e.g. Semantic Model 	e.g. Business Process Model 	e.g. Business Logistics System 	e.g. Work Flow Model 	e.g. Master Schedule 	e.g. Business Plan 	BUSINESS MODEL (CONCEPTS)
Owner	Ent = Business Entry Rel = Business Relationship	Proc = Business Process IO = Business Resources	Node = Business Location Link = Business Linkage	People = Organization Unit Work = Work Product	Time = Business Event Cycle = Business Cycle	End = Business Objective Means = Business Strategy	Executive/Leaders
SYSTEM MODEL (LOGIC)	e.g. Logical Data Model 	e.g. Application Architecture 	e.g. Distributed System Architecture 	e.g. Human Interface Architecture 	e.g. Processing Structure 	e.g. Business Rule Model 	SYSTEM MODEL (LOGIC)
Designer	Ent = Data Entry Rel = Data Relationship	Proc = Application Function IO = User Views	Node = I/O Function (Processor, Storage, etc.) Link = Link Characteristics	People = Role Work = Deliverable	Time = System Event Cycle = Processing Cycle	End = Resource Assertion Means = Action Assertion	Architect
TECHNOLOGY MODEL (PHYSICS)	e.g. Physical Data Model 	e.g. System Design 	e.g. Technology Architecture 	e.g. Presentation Architecture 	e.g. Control Structure 	e.g. Rule Design 	TECHNOLOGY MODEL (PHYSICS)
Builder	Ent = Segment/Table/etc. Rel = Partner/Key/etc.	Proc = Computer Function IO = Data Elements/Sets	Node = Hardware/Systems/Software Link = Link Specifications	People = User Work = Screen Format	Time = Execute Cycle = Component Cycle	End = Condition Means = Action	Engineers
DETAILED REPRESENTATIONS (OUT-OF-CONTEXT)	e.g. Data Definition 	e.g. Program 	e.g. Network Architecture 	e.g. Security Architecture 	e.g. Timing Definition 	e.g. Rule Specification 	DETAILED REPRESENTATIONS (OUT-OF-CONTEXT)
Sub-Constructor	Ent = Field Rel = Address	Proc = Language Statement IO = Control Block	Node = Address Link = Protocol	People = Identity Work = Job	Time = Interval Cycle = Machine Cycle	End = Sub-condition Means = Step	Implementors
FUNCTIONING ENTERPRISE	e.g. DATA	e.g. FUNCTION	e.g. NETWORK	e.g. ORGANIZATION	e.g. SCHEDULE	e.g. STRATEGY	FUNCTIONING ENTERPRISE

© 1986 - 2008 John A. Zachman, Zachman International

Figura 2.7: El Framework de Zachman[Zachman08]

2.3.3.2 The Open Group Architecture Framework (TOGAF)

El framework del Open Group es más conocido por su acrónimo, TOGAF. La visión de TOGAF de una arquitectura empresarial es mostrada en la Figura 2.8.

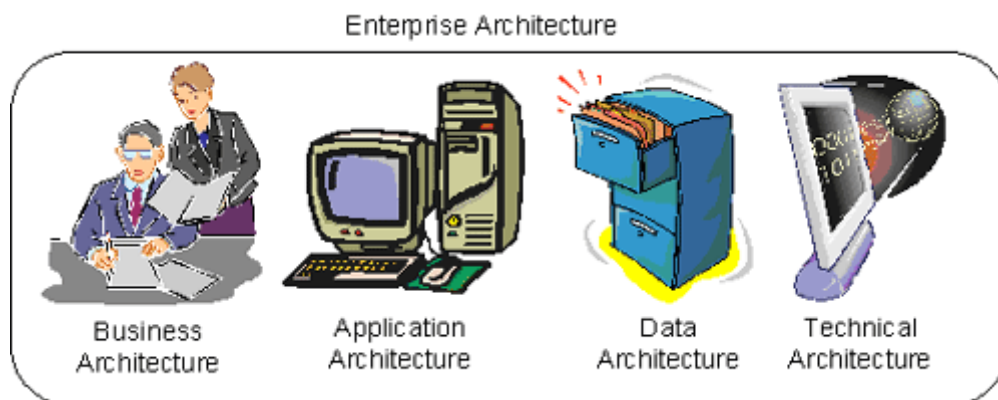


Figura 2.8: Arquitectura empresarial TOGAF[Sessions07]

Como se ve en la figura, TOGAF divide la arquitectura en cuatro categorías:

- Arquitectura del negocio: Describe los procesos del negocio para conocer sus objetivos.

- Arquitectura de aplicación: Describe como las aplicaciones son diseñadas y como interactúan con otras.
- Arquitectura de datos: Describe como los repositorios de datos de la empresa son organizados y accedidos.
- Arquitectura técnica: Describe la arquitectura de hardware y software de soporte a las aplicaciones y sus interacciones.

2.3.3.3 Modelo de vistas “4+1”

En 1995 Philippe Kruchten propuso su célebre modelo “4+1”, ahora vinculado al Rational Unified Process (RUP), que define cuatro vistas diferentes de la arquitectura de software como se ve en la figura 2.9.

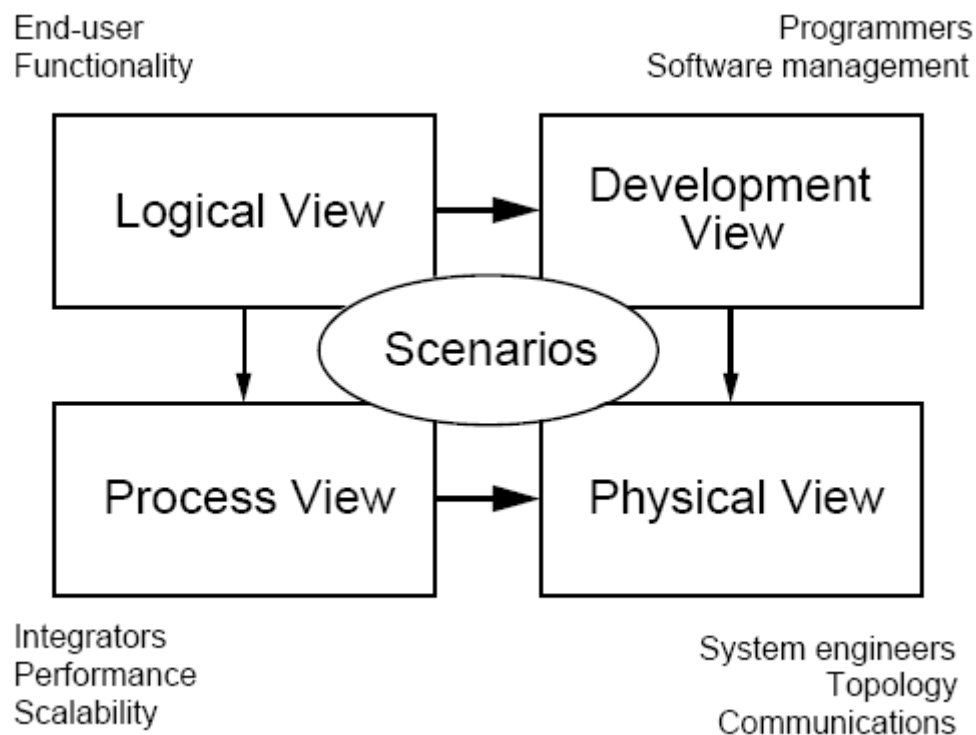


Figura 2.9: El modelo de vistas “4+1” de Kruchten[Kruchten95]

- La vista lógica: Soporta el análisis y la definición de los requisitos funcionales. Comprende las abstracciones fundamentales del sistema a partir del dominio del problema, en la forma de objetos o clases de objetos.
- La vista de proceso: Se tratan algunos requisitos no funcionales como ejecución, disponibilidad, tolerancia a fallos, integridad, etc. La vista se centra en la concurrencia y distribución de procesos.

- La vista física: Se hace un mapeado del software sobre el hardware. Toma en cuenta principalmente los requerimientos no funcionales del sistema como son disponibilidad, tolerancia a fallos, rendimiento y escalabilidad.
- La vista de desarrollo: Aquí se ve la organización estática de módulos en el entorno de desarrollo. El software es empaquetado en módulos o subsistemas que pueden ser desarrollado por una persona o por un pequeño número de desarrolladores.
- Los escenarios: Corresponde a instancias de casos de uso que unifican todas las vistas. Los escenarios son una abstracción de los requerimientos más importantes.

2.3.4 Patrones de Arquitectura

Muchas de las aplicaciones en las que se trabaja se han basado alrededor de un pequeño número de arquitecturas probadas y bien comprendidas, y hay una buena razón para esto, funcionan. Si usamos soluciones conocidas, minimizamos el riesgo de que una aplicación falle debido a una arquitectura inadecuada.

Es así que el primer paso a la hora del diseño es seleccionar un patrón arquitectónico que parezca que satisface los requerimientos principales. A continuación vamos a explicar los principales patrones usados de arquitectura mencionados en [Gorton06].

2.3.4.1 Cliente Servidor N-Capas

La anatomía de este patrón para una aplicación Web se muestra en la figura 2.10.

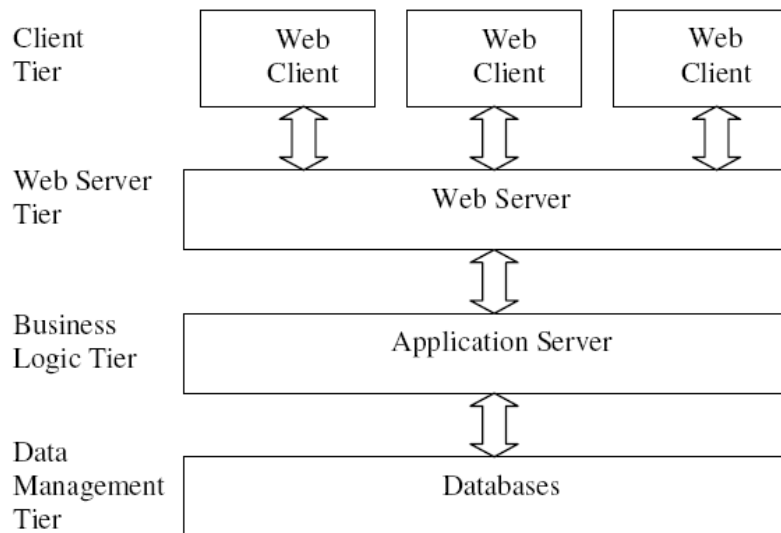


Figura 2.10: Patrón Cliente Servidor N-Capas para una aplicación Web [Gorton06]

Las principales propiedades de este patrón son:

- **Separación de asuntos**, la lógica de presentación, negocios y datos se encuentra en capas separadas.
- **Comunicación síncrona**, de tipo request-reply. Cada capa espera una respuesta de la otra antes de continuar.
- **Despliegue flexible**, todas las capas pueden correr en la misma maquina, o en el otro extremo, cada capa puede ser desplegada en su propia maquina.

La siguiente tabla 2.2 muestra como se manejan los atributos de calidad con este patrón.

Atributo de Calidad	Descripción
Disponibilidad	Los servidores en cada capa pueden ser replicados, de manera que si uno falla otros permanecen disponibles. En general la aplicación ofrecerá un servicio de menor calidad hasta que el servidor sea restaurado.
Manejo de errores	Existe transparencia en la recuperación de fallos. Esto quiere decir que un cliente puede ser redirigido a una replica que puede satisfacer su solicitud.
Capacidad de ser modificado	La separación en capas de presentación, negocios y datos aumenta la capacidad para ser modificado ya que cada capa está claramente encapsulada.
Rendimiento	Esta arquitectura ha demostrado un alto rendimiento. Las cuestiones claves a considerar son la cantidad de hilos simultáneos que soporta cada servidor, la velocidad de las conexiones entre los niveles y la cantidad de datos que se transfieren.
Escalabilidad	Como los servidores en cada capa pueden replicarse y múltiples instancias del servidor se ejecutan en el mismo o en diferentes servidores, existe un buen nivel de escalabilidad.

Tabla 2.2: Atributos de calidad para el patrón Cliente Servido N-Capas

Cabe resaltar que la forma en que cada atributo de calidad es manejado depende de la Web y de la tecnología del servidor de aplicaciones usado para implementar la aplicación.

2.3.4.2 Mensajería

Los principales componentes de este patrón se muestran en la figura 2.11.

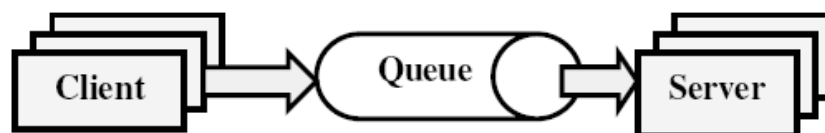


Figura 2.11: Anatomía del patrón de mensajería [Gorton06]

Las principales características de este patrón son:

- **Comunicación asincrónica**, el cliente envía solicitudes a la cola donde son almacenadas hasta la que la aplicación lo remueva. El cliente continúa su trabajo sin esperar que el mensaje sea removido.
- **Calidad de servicio (QoS) configurable**, la cola puede ser configurada para una entrega de alta velocidad poco fiable, o para una entrega fiable y más lenta.
- **Acoplamiento débil**, no hay un enlace directo entre clientes y servidores.

La siguiente tabla 2.3 muestra como los atributos de calidad comunes pueden ser tratados con este patrón.

Atributo de Calidad	Descripción
Disponibilidad	Colas físicas con el mismo nombre lógico pueden ser replicadas sobre diferentes instancias del servidor de mensajería.
Manejo de errores	Si un cliente se comunica con una cola que falla, puede encontrar una replica de la cola y colocar el mensaje allí.
Capacidad de ser modificado	El acoplamiento débil promueve un alto nivel de modificabilidad. Los cambios en los formatos del mensaje de los clientes pueden causar cambios en la implementación del servidor.
Rendimiento	La tecnología de la gestión de colas de mensajes puede entregar cientos de mensajes por segundo. La mensajería poco fiable es más rápida que la fiable.
Escalabilidad	Las colas pueden ser alojadas en los puntos finales de comunicación o ser replicadas a través de clusters de servidores de mensajería hospedados en un único servidor o en múltiples máquinas. Esto hace que la mensajería sea una solución altamente escalable.

Tabla 2.3: Atributos de calidad para el patrón de mensajería.

Es importante nuevamente recalcar que el soporte exacto para estos atributos de calidad es dependiente del producto de mensajería.

El patrón de mensajería es apropiado especialmente cuando el cliente no necesita una respuesta inmediata después de enviar una solicitud.

2.3.4.3 Broker

Los elementos principales del patrón broker son mostrados en la figura 2.12.

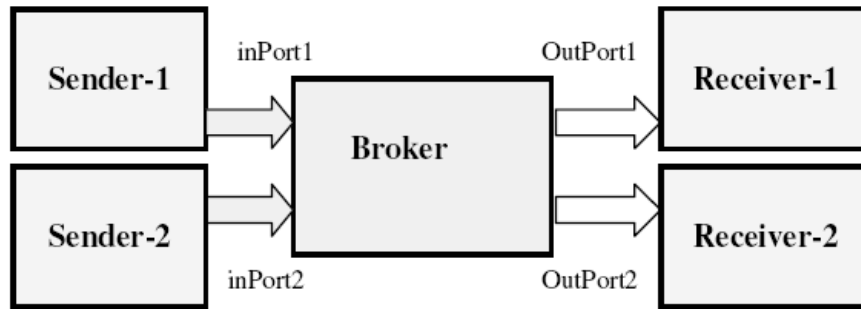


Figura 2.12: Elementos del patrón Broker [Gorton06]

Las principales características de este patrón son:

- **Arquitectura hub-and-spoke**, las conexiones al broker son vía puertos que están asociados con un formato de mensaje específico.
- **Realiza enrutamiento de mensajes**, el broker contiene lógica de procesamiento para entregar un mensaje recibido en un puerto de entrada hacia un puerto de salida.
- **Realiza transformación del mensajes**, la lógica del broker transforma el tipo de mensaje recibido a un tipo de mensaje de destino en el puerto de salida.

La siguiente tabla 2.4 muestra cómo los atributos de calidad pueden ser manejados con este patrón.

Atributo de Calidad	Descripción
Disponibilidad	Los brokers deben ser replicados para asegurar la disponibilidad.
Manejo de errores	Los brokers pueden validar y descartar cualquier mensaje que se les haya enviado en un formato erróneo.
Capacidad de ser modificado	Los brokers separan la transformación y la lógica del enrutamiento del mensaje, de los emisores y receptores. La transformación y la lógica de enrutamiento se pueden modificar sin afectar a éstos.
Rendimiento	Los brokers pueden convertirse potencialmente en un cuello de botella, especialmente si deben dar servicio a un gran volumen de mensajes y ejecutar transformaciones complejas.
Escalabilidad	Se pueden agrupar instancias de brokers para manejar altas cargas de solicitudes.

Tabla 2.4: Atributos de calidad para el patrón Broker

Los brokers son adecuados para aplicaciones en las que el intercambio de mensajes requiere una amplia transformación entre los formatos del emisor y receptor.

2.3.4.4 Coordinador de Procesos

Los componentes del patrón coordinador de procesos se muestran en la figura 2.13.

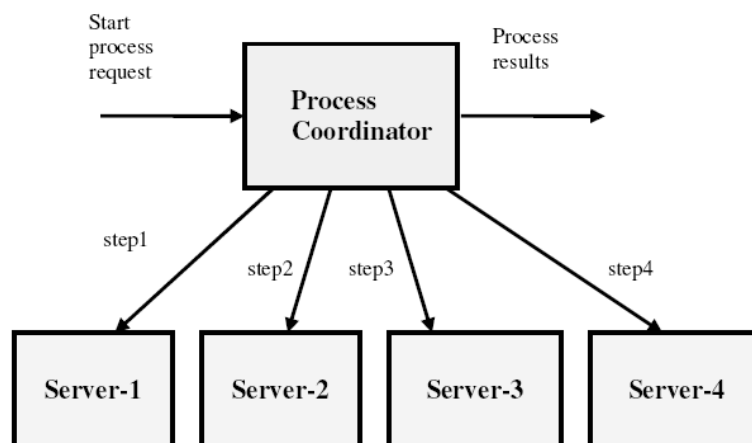


Figura 2.13: Componente del patrón coordinador de procesos [Gorton06]

Las principales características de este patrón son:

- **Los procesos son encapsulados**, el coordinador de procesos encapsula la secuencia de pasos necesarios para cumplir los procesos de negocio.

- **Acoplamiento débil**, los servidores desconocen su rol en el proceso de negocio y el orden de los pasos en el proceso. Los servidores simplemente definen una serie de servicios que pueden realizar y el coordinador los llama cuando sea necesario como parte del proceso de negocio.
- **Comunicaciones flexibles**, pueden ser síncronas o asíncronas.

La tabla 2.5 muestra como los atributos de calidad pueden ser manejados con este patrón.

Atributo de Calidad	Descripción
Disponibilidad	El coordinador, por ser un único punto de falla, tiene que replicarse para crear una solución de alta disponibilidad.
Manejo de errores	El manejo de errores es complejo ya que puede ocurrir en cualquier escena en la coordinación del proceso de negocio. Para manejar los fallos se necesita un diseño cuidadoso para asegurar que los datos contenidos en los servidores sigan siendo coherentes.
Capacidad de ser modificado	La capacidad de ser modificado es mayor debido a que la definición del proceso se encapsula en el coordinador. Los servidores pueden cambiar su implementación sin afectar al coordinador y a los otros servidores siempre y cuando la definición externa de su servicio no cambie.
Rendimiento	Para lograr un alto rendimiento, el coordinador debe manejar solicitudes múltiples y concurrentes, y manejar el estado de cada solicitud además de su progreso a través del proceso
Escalabilidad	El coordinador puede ser replicado.

Tabla 2.5: Atributos de calidad para el patrón coordinador de procesos.

Este patrón es usado comúnmente para implementar procesos de negocio complejos que deben manejar solicitudes de diferentes componentes de servidor. Las tecnologías de los brokers de mensajes y los orquestadores de procesos de negocio están diseñados especialmente para apoyar este patrón.

2.3.5 Otros conceptos relacionados con la Arquitectura: Patrones Arquitectónicos, Modelos de Referencia o Frameworks y Arquitecturas de Referencia

Estos tres conceptos no son arquitecturas, son conceptos muy útiles que capturan los elementos de una arquitectura. La relación entre estos tres conceptos es dada en [Bass+03] y se muestra en la figura 2.14.

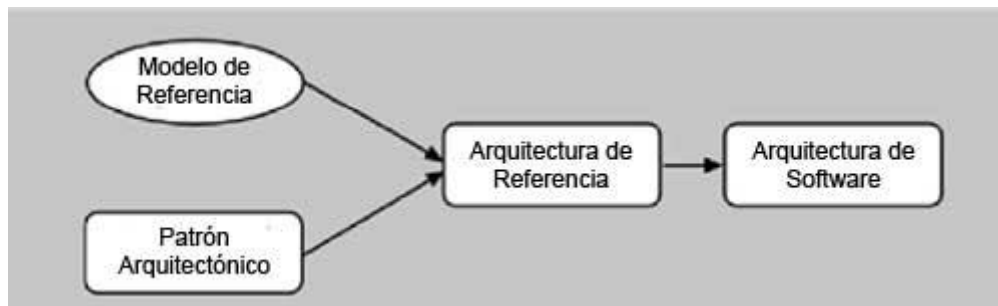


Figura 2.14: Relación entre modelos de referencia, patrones arquitectónicos, arquitectura de referencia y arquitectura de software. [Bass+03]

A continuación procedemos a definir cada uno de ellos.

2.3.5.1 Patrón Arquitectónico

Es una descripción de los elementos y tipos de relación junto con un conjunto de restricciones en la forma que pueden ser utilizados. Un patrón puede ser pensado como un conjunto de restricciones en una arquitectura – en los tipos de elementos y sus patrones de interacción – y éstas restricciones definen un conjunto o familia de arquitecturas que las satisfacen. Un patrón arquitectónico no es una arquitectura, pero transmite una imagen muy útil del sistema, impone limitaciones muy útiles en la arquitectura y, a su vez, en el sistema.

Uno de los aspectos más útiles de los patrones es que ellos exhiben atributos de calidad conocidos. Esto es por qué un arquitecto escoge un patrón particular y no uno al azar. Algunos patrones representan soluciones conocidas para problemas de rendimiento, otros se prestan mejor para los sistemas de alta seguridad, mientras que otros se han utilizado con éxito en sistemas de alta disponibilidad.

El término estilo arquitectónico ha sido ampliamente usado para describir el mismo concepto.

2.3.5.2 Modelo de Referencia o Framework

Un modelo de referencia es una división de la funcionalidad junto con el flujo de datos entre las piezas. Un modelo de referencia es una descomposición estándar de un problema conocido, en partes que cooperativamente resuelven el problema.

2.3.5.3 Arquitectura de Referencia

Una arquitectura de referencia es un modelo de referencia mapeado en elementos de software (que cooperativamente implementan la funcionalidad definida en el modelo de referencia) y el flujo de datos entre ellos. Mientras que un modelo de referencia divide la funcionalidad, una arquitectura de referencia es el mapeo de esa funcionalidad en una descomposición del sistema.

2.4 Arquitectura Orientada a Servicios (SOA) y Web Services

Los problemas de integración tienen un grado de complejidad. Esta complejidad se puede reducir dividiendo el soporte de TI en componentes, donde cada uno de ellos tenga un propósito específico y bien definido, de tal manera que pueda ser reacomodado o reutilizado dependiendo de donde se necesite su funcionalidad. El concepto de SOA parte de esta definición.

“SOA es una arquitectura de aplicación en la cual todas las funciones se definen como servicios independientes con interfaces invocables bien definidas, que pueden ser llamadas en secuencias definidas para formar procesos de negocio.” [Reynoso]

En un ambiente con muchos sistemas especializados, la arquitectura define como se interconectan los sistemas y juntos funcionan como soporte de TI para el negocio. La arquitectura determina qué sistemas y componentes existen en el ambiente y como se da el intercambio de información entre ellos. Como SOA es una arquitectura, entonces no se encuentra atada a ningún lenguaje de programación, vendedor o producto.

El principal objetivo de SOA es alinear el mundo de los negocios con el mundo de la TI de tal manera que sean más efectivos. Es por eso que SOA se apoya del modelo del negocio que es donde se describen los procesos, la estructura organizacional, los objetivos y metas a corto y largo plazo, las influencias económicas y del mercado que afectan a como el negocio logra sus metas, y las reglas y políticas que condicionan

cómo opera el negocio. Lo que diferencia a SOA de otras iniciativas de arquitectura de software es que está diseñado para imitar el flujo de los procesos de negocio.

Las definiciones de SOA dentro de una organización varían, dependiendo del rol y el contexto, en cuatro perspectivas [IBM]:

- Desde una perspectiva de negocios, SOA define un conjunto de servicios de negocio compuestos, que capturan el diseño del negocio de la organización tanto interna (personal) como externamente (clientes y proveedores).
- Desde una perspectiva arquitectónica, SOA es un estilo de arquitectura que soporta orientación a servicios.
- Desde un nivel de implementación, SOA es realizado usando una infraestructura basada en estándares, modelos de programación y tecnologías como los Web Services.
- Desde una perspectiva operacional, SOA incluye un conjunto de relaciones y acuerdos entre consumidores de servicio y proveedores.

2.4.1 ¿Qué es un servicio?

Desde una perspectiva de negocio, los servicios son recursos de IT que corresponden a las actividades del negocio del mundo real o funciones de negocio reconocibles y que se pueden acceder de acuerdo a las políticas de servicio que se han establecido para éstos. Las políticas de servicio definen quién o qué está autorizado a acceder al servicio, cuando el servicio está disponible, el costo de usar el servicio, los niveles de fiabilidad para el servicio, los niveles de rendimiento para el servicio y así sucesivamente.

Desde un punto de vista técnico, los servicios son recursos de IT de granularidad gruesa, reutilizables, que tienen interfaces bien definidas que separan claramente el acceso externo de la interfaz de los servicios de su implementación técnica. Esta separación de la interfaz y la implementación separa a los solicitantes de los proveedores de servicios a fin de que ambas puedan evolucionar independientemente.

Las principales características de un servicio son [Colan04]:

- *Acoplamiento débil*: El enlace entre un proveedor de servicio y un solicitante debe ser de acoplamiento débil. Esto quiere decir que el solicitante no tiene conocimiento de los detalles técnicos de la implementación del proveedor, como

el lenguaje de programación, la plataforma de despliegue, etc. Las invocaciones de servicio se hacen por medio de mensajes, en lugar de utilizar APIs o formatos de archivos. Este acoplamiento permite que el software a cada lado de la conversación cambie sin afectar a los demás, con la condición de que el mensaje mantenga el mismo esquema.

- *Interfaces bien definidas*: La interacción de los servicios debe ser bien definida. Web Services Description Language (WSDL) es una manera de describir los detalles requeridos por un solicitante del servicio para enlazarse a un proveedor de servicio. WSDL no incluye ningún detalle de la tecnología de la implementación del servicio. El servicio solicitante no sabe ni le importa si el servicio está escrito en un lenguaje de programación dado. La definición común de WSDL permite a las herramientas de desarrollo crear interfaces comunes para varios estilos de interacción, mientras que se esconden los detalles de la forma como se invoca el servicio desde el código de la aplicación.
- *Diseño de Servicios sin estado*: Los servicios deben ser independientes, no deben requerir de información o estado de una solicitud a otra cuando es implementado. Los servicios no deben ser dependientes del contexto o estado de otros servicios. Cuando se requieren dependencias, éstas son mejor definidas en términos de procesos de negocio, funciones o modelos de datos. Claro que las aplicaciones solicitantes requieren estados persistentes entre invocaciones de servicio, pero esto debe ser separado del proveedor del servicio.
- *Granularidad del Servicio*: La granularidad de las operaciones es un punto importante del diseño. El uso de interfaces de granularidad gruesa para el consumo externo es recomendada, mientras que las interfaces de granularidad fina pueden usarse dentro de la empresa. Mientras que las interfaces de grano fino ofrecen mayor flexibilidad a la aplicación solicitante, también significa que los patrones de interacción pueden variar entre diferentes solicitantes del servicio. Esto puede hacer que el soporte sea más difícil para el proveedor del servicio. Una interfaces de granularidad gruesa garantiza que los solicitantes usen el servicio de forma coherente. SOA no requiere el uso de interfaces de granularidad gruesa, pero recomienda su uso como una buena práctica para la integración externa. La coreografía de servicios puede usarse para crear una

interfaz de granularidad gruesa que ejecuta un proceso de negocio que consiste de operaciones de granularidad fina.

2.4.2 Los colaboradores de SOA

La figura 2.15 muestra los elementos colaboradores de una arquitectura orientada a servicios. Un solicitante (*Service Consumer*) realiza una ubicación de servicio dinámica consultando el registro (*Service Registry*) para un servicio que se ajuste a sus criterios. Si el servicio existe, el registro proporciona al solicitante la información de la interfaz y la dirección del servicio proveedor (*Service Provider*).

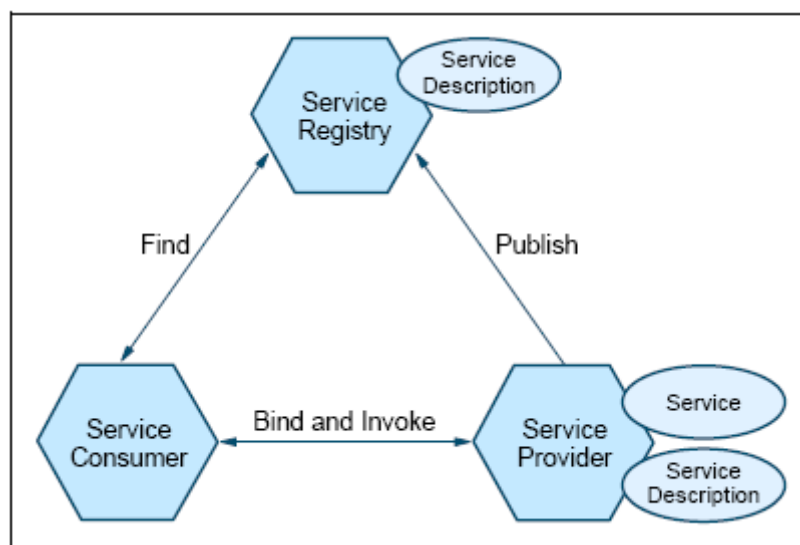


Figura 2.15: Los colaboradores en una arquitectura orientada a servicios.

Los componentes básicos de una arquitectura orientada a servicios son [Endrei+04]:

- **El consumidor del servicio (*Service Consumer*):** Es una aplicación, un módulo u otro servicio que requiere un servicio. El consumidor localiza (descubre) entradas en el registro de servicios (*Service Registry*), y a continuación se enlaza con el proveedor del servicio (*Service Provider*) a fin de invocar el servicio definido.
- **El proveedor del servicio (*Service Provider*):** Es una entidad accesible en la red que acepta y ejecuta solicitudes de los consumidores. El proveedor publica sus interfaces y su información de acceso en el registro de servicios (*Service Registry*). Cada proveedor debe decidir entre otras cosas que servicios exponer y evaluar el equilibrio entre seguridad y la fácil disponibilidad.

- **El registro de servicios (*Service Registry*):** El registro de servicios es el que habilita el descubrimiento de servicios. Contiene un repositorio de los servicios disponibles y permite la búsqueda de las interfaces de los proveedores de servicios (*Service Provider*) a los consumidores interesados. Los que implementa un registro de servicio deben considerar el alcance con el que el registro será implementado. Por ejemplo, hay servicios que son accesibles públicamente mediante Internet y hay otros servicios que son privados y son accesibles a los usuarios de una empresa mediante una intranet.

Cada componente puede actuar también como uno de los dos otros componentes. Por ejemplo si un proveedor de servicio necesita información de otro servicio, puede actuar como consumidor del servicio.

Las operaciones en una arquitectura orientada a servicios son [Endrei+04]:

- **Publicar (*Publish*):** Para ser accesible, una descripción del servicio debe ser publicada para así ser descubierta e invocada por un consumidor de servicio.
- **Encontrar (*Find*):** Un consumidor de servicio localiza un servicio consultando en el registro por un servicio que cumpla con sus criterios.
- **Enlazar e Invocar (*Bind, Invoke*):** Luego de recibir la descripción del servicio, el consumidor procede a invocar el servicio de acuerdo a la información en la descripción del servicio.

2.4.3 El ciclo de vida de SOA

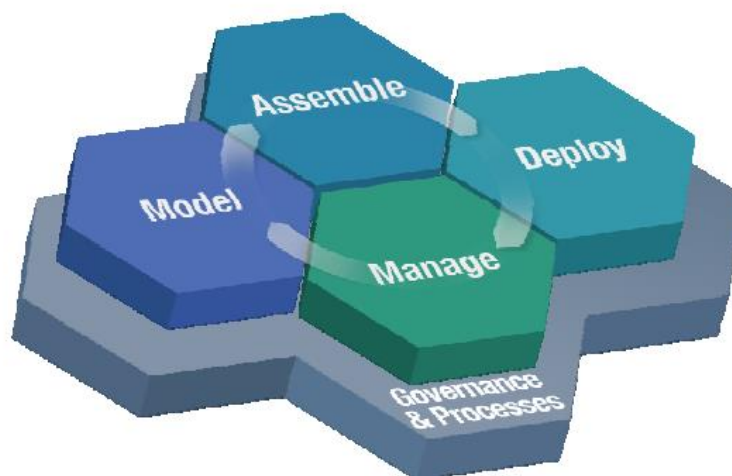


Figura 2.16: El ciclo de vida de SOA [High+05]

El ciclo de vida de SOA, como se muestra en la figura 2.16, incluye las siguientes fases:

- Modelado
- Ensamblaje
- Despliegue
- Administración
- Gobernabilidad y procesos

El ciclo de vida es entonces cubierto por un conjunto de procesos de gobernabilidad que aseguran la conformidad y el cumplimiento de las políticas operacionales, y el cambio se produce de una manera controlada y con la autoridad apropiada tal como lo previsto en el diseño del negocio.

La progresión del ciclo de vida no es totalmente lineal [High+05]. Por ejemplo los cambios en la información clave de rendimiento en la Fase de Modelado necesitan alimentar directamente a la fase de Administración para actualizar el ambiente operacional. Las limitaciones en la fase de Despliegue como la limitación de los recursos pueden condicionar algunas de las decisiones de la Fase de Ensamblaje. Y, ocasionalmente, las limitaciones en la tecnología de la información establecidas en la fase de Ensamblaje limitarán el modelo de negocio creado durante de la fase de Modelado.

2.4.3.1 Modelado

Modelado es el proceso de capturar el diseño del negocio comprendiendo las necesidades y objetivos del negocio. Los requerimientos del negocio se traducen en una especificación de los procesos de negocio, metas y suposiciones para la creación de un modelo de negocio. Muchas empresas no realizan esta práctica. En algunos casos las empresas que realizan el modelado hacen uso de técnicas primitivas de dibujo o utilizan documentos de texto.

Según [High+05] la captura de un diseño de negocio usando un método sofisticado que incluya el uso de una herramienta especializada permite realizar escenarios “what-if” con varios parámetros que la empresa puede experimentar. El proceso también puede ser simulado usando aquellos parámetros para predecir el efecto que el proceso tendrá

en el negocio y en los sistemas de IT. Si los resultados obtenidos no coinciden con los objetivos del negocio, entonces la definición del proceso puede ser refinado.

2.4.3.2 Ensamblaje

El diseño de negocio es usado para comunicar los objetivos del negocio en la organización de IT que ensamblará los artefactos de los sistemas de información que implementan el diseño. El arquitecto de la empresa trabaja en estrecha colaboración con el analista del negocio para convertir el diseño del negocio en un conjunto de definiciones de procesos de negocio, así como las actividades usadas para obtener los servicios requeridos de las definiciones de las actividades. El arquitecto de la empresa y el analista del negocio trabajan con el arquitecto de software para completar el diseño de los servicios.

Durante el proceso de resolver el diseño y la implementación de los procesos de negocio modelados y los servicios, se debe realizar una búsqueda de los artefactos y aplicaciones existentes en un esfuerzo por encontrar los componentes que satisfagan las necesidades del diseño. Algunas aplicaciones se adaptan perfectamente, otras tendrán que ser reconstruidas y algunas tendrán que ser aumentadas para cumplir los requerimientos del diseño.

Por último, la fase de ensamblaje incluye aplicar un conjunto de políticas y condiciones para controlar como las aplicaciones operan en una ambiente de producción en tiempo de ejecución. Además la fase de ensamblaje incluye características operacionales críticas como la localización de las restricciones, dependencia de recursos, control de integridad y protección de acceso.

2.4.3.3 Despliegue

La fase de despliegue del ciclo de vida incluye una combinación de crear el ambiente de hospedaje para las aplicaciones y las tareas de despliegue de esas aplicaciones. Esto incluye resolver las dependencias de los recursos de las aplicaciones, condiciones operacionales, requerimientos de capacidad y limitaciones de integridad y acceso.

Una serie de inquietudes son relevantes para la construcción del ambiente de hospedaje incluyendo la presencia de la infraestructura de las aplicaciones existentes y los servicios pre-existentes. Más allá de eso, se deben considerar ofrecer plataformas

adecuadas para la lógica de interacción del usuario, los flujos de proceso de negocio, los servicios de negocio, los servicios de acceso y la información lógica.

2.4.3.4 Administración

Comprende las tareas, tecnología y software usado para administrar y supervisar los recursos de la aplicación es decir, servicios y procesos de negocio, que son desplegados en un ambiente de producción.

La supervisión es un elemento fundamental para asegurar que los sistemas están levantados y ejecutándose para mantener los requerimientos de disponibilidad de los servicios del negocio. También se requiere supervisar el rendimiento de los consumidores de servicio y el tiempo de respuesta de los proveedores. Además la supervisión incluye mantener logs de problemas para detectar fallas en varios servicios y componentes, así como localizar esas fallas y restaurar el estado operacional del sistema.

La administración también implica la realización de rutinas de mantenimiento, administrar y asegurar las aplicaciones, recursos y usuarios; y la predicción del crecimiento futuro para garantizar que los recursos estarán disponibles en el momento que se demanden.

2.4.3.5 Gobernabilidad y Procesos

La gobernabilidad ayuda a los clientes a extender los planes de SOA en toda la empresa de una manera controlada. Esta nueva capacidad de gobernabilidad de SOA ayuda a los clientes a establecer una línea base para medir mejoras, hacer seguimiento a los proyectos SOA, construir un conjunto de recursos calificados y establecer la estructura para la toma de decisiones sobre iniciativas SOA. Además esta capacidad de los servicios ayuda a mantener todas las iniciativas SOA, arquitecturas e inversiones alineadas con los objetivos del negocio.

Un aspecto clave de la gobernabilidad es que se definen las normas, los procesos, las métricas y las construcciones organizacionales para apoyar la planificación, la toma de decisiones, la dirección, y el control de la IT y SOA para lograr los objetivos del negocio.

2.4.4 La Arquitectura de Referencia SOA

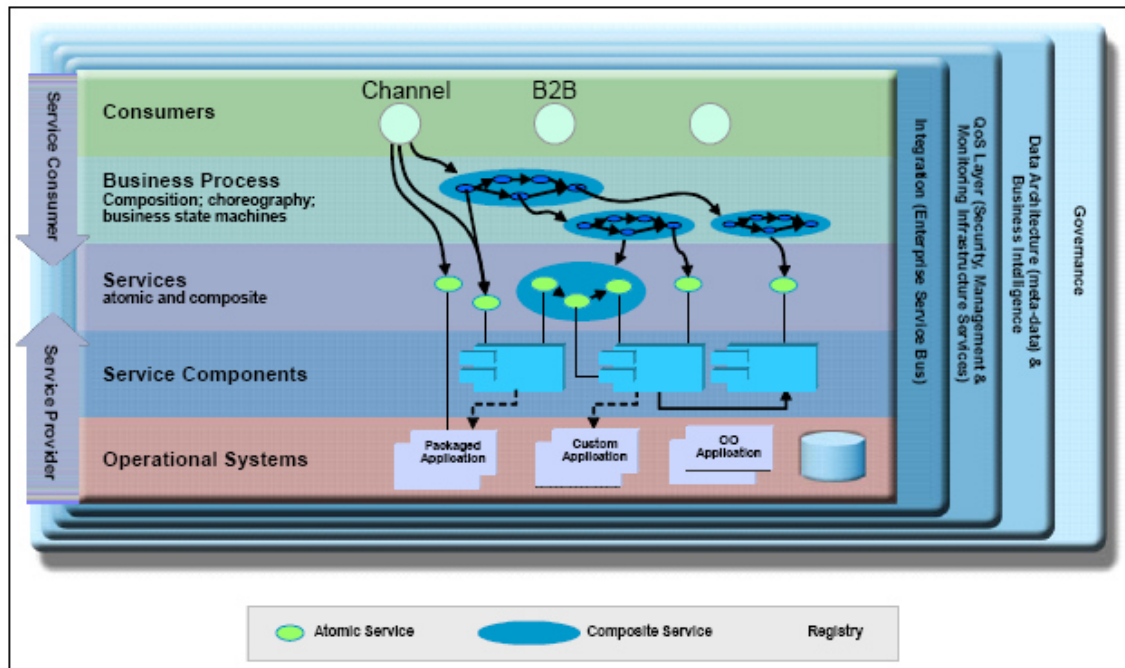


Figura 2.17: Una vista de la solución de la Arquitectura SOA[Wahli+07]

La figura 2.17 muestra una arquitectura de referencia para descomponer un diseño SOA. Para SOA el rol del arquitecto del negocio es muy importante ya que es el responsable de abarcar tanto el diseño del negocio como los sistemas de información que codifican ese diseño.

Si tomamos un enfoque “top-down”, el arquitecto comienza por la identificación de los procesos de negocio y servicios de negocio utilizados por los usuarios de la empresa. Los usuarios de la empresa son los consumidores de los procesos y servicios. Los procesos de negocio pueden ser composiciones de otros procesos o servicios, por lo tanto deben ser descompuestos en sub-procesos y servicios.

Los procesos de negocio y servicios son entonces detallados en componentes de servicios. Los componentes de servicios incluyen un conjunto detallado de definición de metadatos utilizados para describir el servicio.

El conjunto resultante de las definiciones de los procesos de negocio, servicios y esquemas constituyen la arquitectura lógica de la aplicación. El arquitecto entonces debe mapear esa arquitectura lógica a una arquitectura física.

2.4.5 Modelo de la Arquitectura Lógica SOA

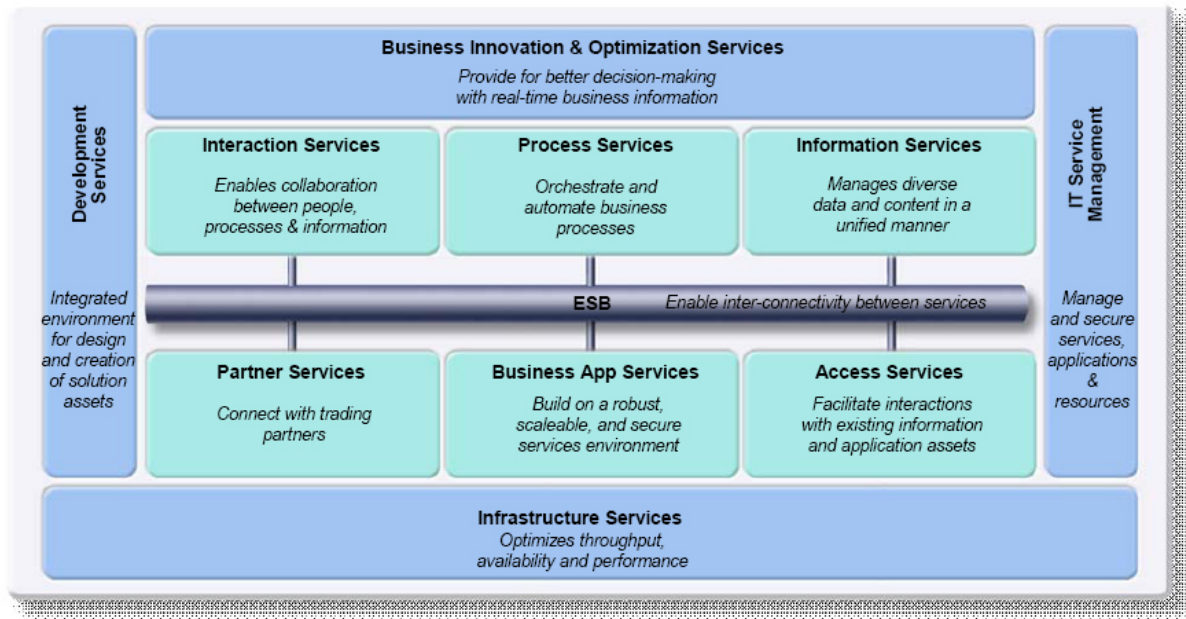


Figura 2.18: Arquitectura Lógica de SOA[Wahli+07]

A continuación vamos a hacer una breve descripción de cada uno de los tipos de servicios que se encuentran en la arquitectura lógica mostrada en la figura 2.18. Los servicios de las cajas de color verde, son los principales ya que son el conjunto básico utilizado por la aplicación en tiempo de ejecución en el momento del despliegue. Los servicios restantes son usados para dar soporte a los servicios principales.

Los servicios principales según [Wahli+07] son los siguientes:

- **Servicios de interacción:** Proporcionan la capacidad requerida para entregar funciones de IT y datos a los usuarios, conociendo sus preferencias específicas.
- **Servicios de proceso:** Proporcionan la capacidad de control requerida para manejar el flujo y las interacciones de múltiples servicios de manera que implementen procesos de negocios.
- **Servicios de aplicación de negocio:** Son llamados por los consumidores de servicio. Implementan la lógica principal del negocio.
- **Servicios de información:** Proporcionan las capacidades necesarias para federar, replicar y transformar fuentes de datos dispares.

- **Servicios de acceso:** Proporciona las capacidades para integrar aplicaciones legadas y funciones en un SOA. En otras arquitecturas a veces es referido como adaptadores.
- **Servicios asociados:** Proporcionan el documento, protocolo y la administración de capacidades de los socios para los procesos de negocio que implican la interacción con proveedores y socios externos.

Los servicios de soporte según [Wahli+07] son:

- **Enterprise Service Bus (ESB):** Su presencia en la arquitectura es transparente a los servicios de aplicación SOA. Este bus proporciona una infraestructura que remueve la dependencia de la conexión directa entre los consumidores y proveedores de servicio. Los consumidores se conectan al bus y no al proveedor que implementa el servicio.
- **Servicios de Innovación y Optimización del negocio:** Son principalmente usados para representar las herramientas y las estructuras de metadatos que codifican el diseño del negocio incluyendo las políticas y objetivos. Ayudan a capturar, codificar, analizar y refinar de manera iterativa el diseño del negocio.
- **Servicios de desarrollo:** Abarcan toda la suite de herramientas de arquitectura, herramientas de desarrollo, herramientas de composición visual, herramientas de ensamblaje, metodologías, etc. para construir una aplicación basada en SOA.
- **Administración de servicios de IT:** Representa el conjunto de herramientas de administración usadas para monitorear los flujos de servicio, la utilización de recursos, la identificación de cuellos de botella, el logro de los objetivos del servicio, el cumplimiento de las normas administrativas y la recuperación de fallos.
- **Servicios de Infraestructura:** Forman el núcleo del ambiente de ejecución de la tecnología de información usada para hospedar aplicaciones SOA. Estos servicios ofrecen la capacidad de optimizar el rendimiento, la disponibilidad y la gestión.

2.4.6 Las tecnologías de los Web Services

Los Web Services es una tecnología que ha recibido una amplia aceptación con la implementación de la arquitectura orientada a servicios. Esto se debe a que los Web Services proporcionan un enfoque de computación distribuida para integrar aplicaciones extremadamente heterogéneas sobre Internet. Las especificaciones de los Web Services son completamente independientes del lenguaje de programación, sistema operativo y hardware para promover el acoplamiento débil entre el consumidor del servicio y el proveedor. La tecnología se basa en estándares abiertos como XML, SOAP, UDDI y WSDL que se describen a continuación.

2.4.6.1 Extensible Markup Language (XML)

XML es un lenguaje de marcas que subyace en la mayoría de las especificaciones usadas en los Web Services. XML es un lenguaje genérico que puede utilizarse para describir el contenido en una forma estructurada, separada de su presentación en un dispositivo específico.

2.4.6.2 Simple Object Access Protocol (SOAP)

SOAP es una especificación para el intercambio de mensajes estructurados basados en XML, entre el proveedor, consumidor y el registro de servicios, que consiste de 3 partes:

- El formato de un mensaje SOAP es un *sobre* conteniendo cero o más *cabezeras* y un *cuerpo*. El *sobre* es el elemento superior del documento XML, ofreciendo un contenedor para el control de la información, el destinatario del mensaje, y el mensaje en sí. Las *cabezeras* contienen información de control como los atributos de la calidad de servicio. El *cuerpo* contiene la identificación del mensaje y sus parámetros. En la figura 2.19 se muestra una representación de un mensaje SOAP.

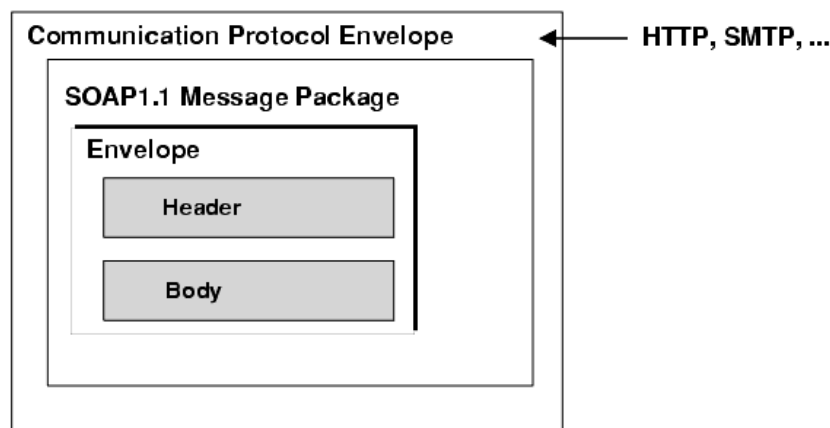


Figura 2.19: Un mensaje SOAP

- Normas codificadas que se usan para expresar instancias de los tipos de datos definidos para la aplicación. SOAP define un esquema de tipos de datos independientes del lenguaje de programación basado en un XML Schema Descriptor (XSD).
- Representación RPC que es la convención para representar llamadas a procedimientos remotos (RPC) y las respuestas.

SOAP, en un principio, es un protocolo de transporte independiente. En consecuencia, puede ser utilizado en combinación con una variedad de protocolos como HTTP, JMS, SMTP o FTP. Actualmente, el modo más común de intercambio de mensajes SOAP es a través de HTTP.

2.4.6.3 Universal Description, Discovery and Integration (UDDI)

Uno de los componentes fundamentales de una arquitectura orientada a servicios es un mecanismo de descripción de Web Services para ser descubierto por los posibles solicitantes. Para establecer esta parte del framework de los Web Services, se requiere un directorio central para hospedar descripciones de servicios. Ese directorio puede ser integrante de una organización, como se ve en la figura 2.20, o de una comunidad de Internet.

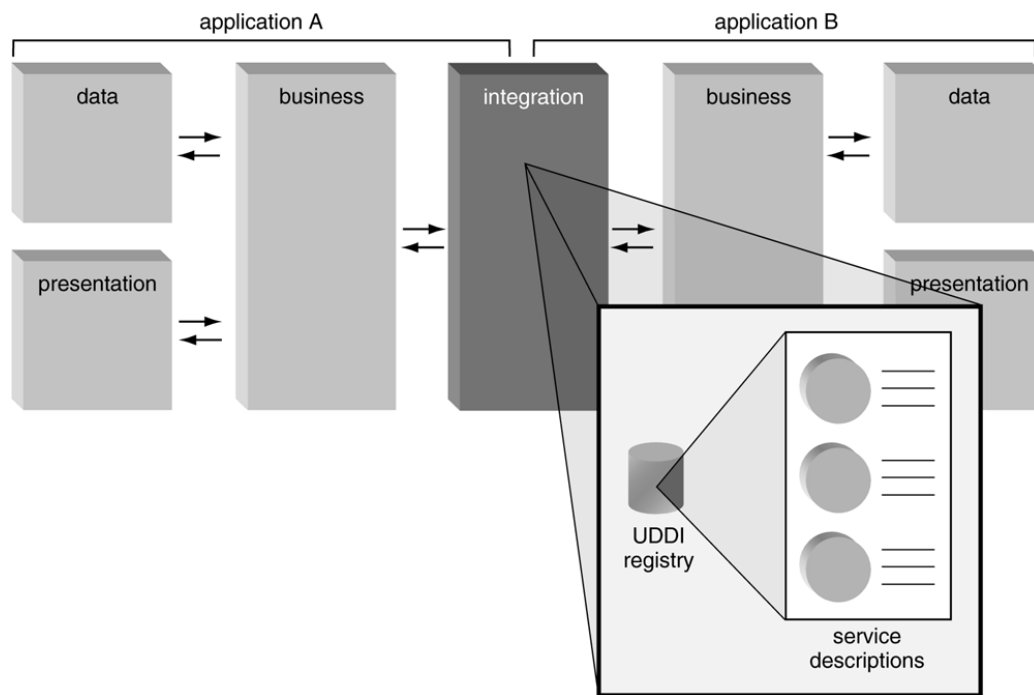


Figura 2.20: Descripciones de servicios centralizados en un registro UDDI [Erl04].

UDDI implementa un registro de servicios. El registro es un intermediario entre el consumidor que necesita encontrar un servicio y los proveedores que publican su disponibilidad en un UDDI. El resultado es como tener un directorio electrónico de páginas amarillas para descubrir los servicios que se necesita para construir una aplicación SOA.

2.4.6.4 Web Service Definition Language (WSDL)

La especificación WSDL define un vocabulario XML para definir un Web Service. Un documento WSDL está definido en cuatro partes:

- Definición de la interfaz: Aquí se definen el puerto de conexión, las operaciones que pueden ser solicitadas y los mensajes que pueden ser usados. En otras palabras, la descripción WSDL define que puedes hacer y cómo hacerlo después de conectarte al puerto.
- Definición del mensaje: Aquí se ve las definiciones de los ítems de datos para cada una de las operaciones que están definidas bajo la especificación de la interfaz. Estas definiciones actúan como templates para que el programa solicitante haga las consultas y para que entienda las respuestas que recibe.

- Definición de tipos: Aquí se definen los tipos de datos que son usados por el Web Service.
- Definición del enlace: Define exactamente como dos programas pueden conectarse.

2.4.7 Enterprise Service Bus (ESB)

El concepto del Enterprise Service Bus no es un producto, sino una mejor práctica para implementar una arquitectura orientada a servicios. Como se muestra en la figura 2.21, éste establece un bus de mensajes de la clase de negocio que combina la infraestructura de mensajería con la transformación de mensajes y el ruteo basado en contenido en una capa de integración lógica entre servicios consumidores y proveedores [Endrei+04].

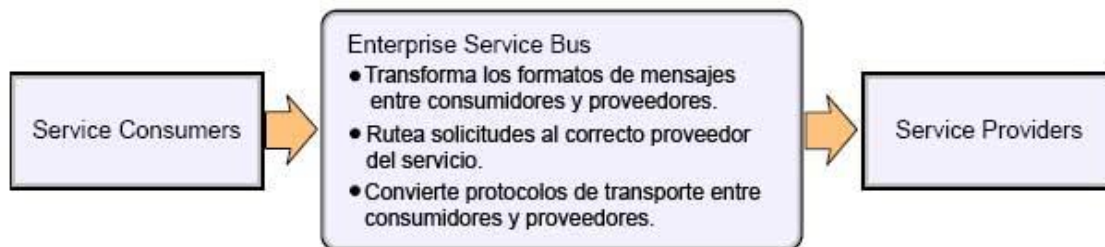


Figura 2.21: Enterprise Service Bus [Endrei+04]

El principal objetivo del Enterprise Service Bus es proporcionar la virtualización de los recursos del negocio, permitiendo que la lógica del negocio de la empresa sea desarrollada y administrada independientemente de la infraestructura, red y disposición de los servicios de negocio. Los recursos en el ESB son modelados como servicios que ofrecen una o más operaciones de negocios.

Según [Endrei+04], implementar un Enterprise Service Bus requiere un conjunto integrado de servicios middleware para soportar los siguientes estilos de arquitectura:

- Arquitecturas orientadas a servicios, donde las aplicaciones distribuidas están compuestas de servicios granulares y reutilizables con interfaces publicadas, compatibles con estándares y bien definidas.
- Arquitecturas orientadas a mensajes, donde las aplicaciones envían mensajes a través del ESB a aplicaciones que los reciben.
- Arquitecturas orientadas a eventos, donde las aplicaciones generan y consumen mensajes independientemente de otras.

Los servicios middleware proporcionados por un Enterprise Service Bus necesitan incluir [Endrei+04]:

- Un middleware de comunicación que soporte una variedad de paradigmas de comunicación (síncrona, asíncrona, request/reply, one way, call-back), calidad del servicio (seguridad, garantía en la entrega, rendimiento), APIs, plataformas y protocolos estándar.
- Un mecanismo de inyección de procesamiento inteligente de las solicitudes de servicio y respuestas dentro de la red.
- Herramientas basadas en estándares para habilitar la integración rápida de los servicios.
- Un sistema de administración para aplicaciones de acoplamiento débil y sus interacciones.

2.4.8 Beneficios de una Arquitectura Orientada a Servicios

Una arquitectura orientada a servicios puede ser desarrollada usando los recursos existentes. Las organizaciones que enfocan sus esfuerzos de desarrollo en la creación de servicios, usando las tecnologías existentes combinados con enfoques basados en componentes para el desarrollo de software, obtienen varios beneficios mencionados en [Tuggle+03]:

- *Aprovechar los recursos existentes:* Este fue el primero, y más importante de los requerimientos. Un servicio de negocio puede ser construido como la suma de los componentes existentes. El uso de este nuevo servicio solo requiere saber como es su interfaz y su nombre. No es necesario conocer como funciona el servicio internamente y como fluyen los datos a través de los componentes. Esto permite a las organizaciones aprovechar las inversiones actuales, crear servicios desde una conglomeración de componentes contruidos en diferentes maquinas con ambientes y arquitecturas heterogéneas. Los sistemas legados pueden ser accedados y encapsulados vía Web Services.
- *Infraestructura común:* El desarrollo y despliegue de la infraestructura se volverá más consistente a través de todas las aplicaciones empresariales. Los componentes existentes, los nuevos componentes desarrollados y los

componentes comprados a terceros pueden ser consolidados desde un framework SOA.

- *El tiempo de salida al mercado es más rápido:* Si nuestra organización ya posee librerías de Web Services, va a ser mucho mas rápido el tiempo de de salida al mercado ya que los servicios y componentes se pueden reutilizar ocasionando así la reducción del diseño, desarrollo, pruebas y tiempo de instalación.
- *Reducción de costos:* Debido a la evolución de los negocios, el costo de mejorar y crear nuevos servicios se reduce adaptando el framework SOA. La línea de aprendizaje para el equipo de desarrollo también disminuye ya que podrían estar familiarizados con los componentes actuales.
- *Mitigación de riesgos:* Reutilizar los componentes existentes reduce el riesgo de introducir nuevas fallas en el proceso de mejoramiento o creación de nuevos servicios.
- *Mejora continua de procesos de negocio:* SOA permite una representación clara de los flujos de procesos identificados por el orden de los componentes usados en un servicio de negocio particular. Esto proporciona a los usuarios de negocio un ambiente ideal para monitorizar los procesos de negocio. El modelado del proceso se refleja en los servicios de negocio. La manipulación del proceso se logra reorganizando los componentes que constituyen un servicio de negocio. Esto ocasiona un cambio un el flujo del proceso y si monitoreamos los efectos, facilita la mejora continua.
- *Arquitectura centrada en el proceso:* La aplicación se desarrolla para el proceso. El proceso esta dividido en una serie de pasos, cada uno representando un servicio de negocio. Así cada servicio o componente funciona como una sub-aplicación. Estas sub-aplicaciones son unidas para crear un flujo de proceso capaz de satisfacer las necesidades del negocio.

2.5 SOA y Web Services para la Integración

Han surgidos dos enfoques usando XML y Web Services para la integración e interoperabilidad [Newcomer+04]:

- Integración de Web Services (WSI): La aplicación táctica y oportuna de Web Services para resolver los problemas de integración e interoperabilidad.
- Integración orientada a servicios (SOI): La aplicación estratégica y sistemática de Web Services para resolver los problemas de integración e interoperabilidad.

Ambos enfoques se basan en la base de XML, SOAP y WSDL. Ambos métodos utilizan la misma tecnología definida por la plataforma de Web Services, pero solo SOI las aplica en una manera estratégica y sistemática basado en los principios de SOA, incluyendo la definición de un framework de gobernabilidad de SOA y definiendo los procesos de SOA y las mejores prácticas que se llevarán a cabo en un departamento o en toda la organización. Por ejemplo, ambos enfoques podrían utilizar UDDI como un registro de servicios, pero SOI definirá una taxonomía de servicio consistente que satisface los requerimientos de la organización a través de varios proyectos, mientras que con WSI se crea un registro UDDI por proyecto con poca o ninguna consideración a través de múltiples proyectos.

En la siguiente sección explicaremos en más detalle en que consiste la Integración Orientada a Servicios.

2.5.1 La Integración Orientada a Servicios (SOI)

Según [Newcomer+04] la implementación de SOI comienza antes de iniciar el primer proyecto. Durante la fase de inicio:

- Se definen el marco de gobernabilidad SOA, los procesos, las directrices, modelos y herramientas.
- Se utiliza el modelado formal de los servicios de dominio. Estos modelos no tienen que ser exhaustivos, pero deberían identificar los tipos de datos principales, los contratos de servicio, y los procesos que son usados dentro de la organización y son visibles a otras organizaciones. Idealmente, algunos de los datos, servicios y modelos de procesos pueden ser reusados.
- Se define la taxonomía del servicio, así los servicios pueden ser catalogados y clasificados en múltiples proyectos para promover la reutilización futura de servicios.

- Cuando la plataforma de Web Services ofrece múltiples opciones para realizar tareas similares, un único enfoque coherente es seleccionado y aplicado a través de múltiples proyectos. Por ejemplo, la distribución de los contratos de servicios WSDL es estandarizada.

Dentro de este marco, un proyecto SOI típico implica:

- Perfeccionamiento de los modelos de datos existentes para dar cabida a este proyecto de integración.
- Definir contratos de servicios formales más allá de lo que puede ser definido en WSDL para incluir reglas de negocio, políticas de seguridad, políticas de administración, políticas de versiones y políticas de recuperación. Siempre que sea posible, WS-Policy se utiliza para definir y comunicar estas decisiones de política.
- Configurar los envoltorios de los sistemas legados que cumplan con el contrato de servicio (mientras que encapsulan los detalles de la implementación de los sistemas legados), o la creación de nuevos servicios que se requieren para este proyecto de integración.
- Definición de las transformaciones que mapean datos entre diferentes modelos de datos cuando éstos cruzan las fronteras del dominio de los servicios.
- Configuración del ambiente de ejecución para la plataforma de Web Services para ofrecer y hacer cumplir las cualidades de los servicios de las empresas como los requisitos transaccionales, la entrega de mensajes garantizada, y políticas de recuperación.

Capítulo 3: Soluciones para la Integración de Sistemas

En las organizaciones de hoy es frecuente encontrar lo que se denominan islas de información, que no son más que Sistemas de Información desarrollados para satisfacer las necesidades específicas de la organización, muchas veces usando plataformas, lenguajes, bases de datos diferentes y lo más importante, no se encuentran directamente comunicadas. Los sistemas de información han evolucionado dando origen al concepto de Integración de Sistemas.

3.1 Tecnologías Middleware

Un middleware es un software de conectividad que consiste en un conjunto de servicios que permiten a múltiples procesos ejecutarse en una o más máquinas para interactuar a través de una red. Un middleware es esencial para migrar aplicaciones mainframe a cliente/servidor y para proporcionar la comunicación a través de plataformas heterogéneas [Bray97].

3.1.1 Remote Procedure Call (RPC)

Es el middleware diseñado como servicio síncrono para permitir la gestión remota de redes. Esconde las operaciones de envío y recepción bajo el aspecto de una llamada convencional a una rutina o procedimiento. Los RPC tienen la misma semántica que las llamadas a procedimientos ordinarios; es decir, se realiza la llamada y se pasa el control al procedimiento servidor; cuando éste devuelve el resultado, el cliente recupera el control.

Un sistema RPC nos permite [Zaragoza06]:

- Ocultar la distribución detrás de las llamadas a procedimientos.
- Ofrecer un lenguaje de definición de interfaces (IDL) para describir los servicios.
- Generar el código necesario para realizar las llamadas remotas y tratar con todos los aspectos de comunicación.

- Suministrar un “binder” si se cuenta con un directorio de servicios y nombres.

3.1.2 Transaction Processing Monitors (TP Monitors)

Es un Middleware para Procesamiento de Transacciones ya que facilita la conectividad y el acceso a un gran número de usuarios. Este tipo de middleware requiere del soporte de un monitor, es decir, un programa que supervise las transacciones entre procesos, con el propósito de asegurar el éxito de la transacción, o en caso de ocurrir un error, tomar acciones apropiadas. Su principal uso es coordinar el flujo de solicitudes entre los dispositivos y las aplicaciones que procesan esas solicitudes [Mendoza05].

3.1.3 Database Access Technology (DBAT)

Son las APIs que crean una capa transparente para el acceso a base de datos, ocultando la complejidad dada por el manejador de base de datos. Ejemplo de estas API son Java Database Connectivity (JDBC) desarrollado por SUN y Open Database Connectivity (ODBC) desarrollado por Microsoft®.

3.1.4 Distributed Object Middleware (DOM)

Es el Middleware para tecnologías orientadas a objetos; los objetos piden servicio a otros objetos que se encuentran en la red. Se encarga de establecer comunicación entre los clientes y los objetos de forma transparente respecto a la distribución. Permite localizar a un objeto remoto dada una referencia a ese objeto. El núcleo de estos Middleware es Object Request Broker (ORB). Ejemplos de estos Middleware son: Common Object Request Broker Architecture (CORBA) de OMG, Remote Method Invocation (RMI) de SUN Microsystems y Distributed Component Object Model (DCOM) de Microsoft.

3.1.4.1 Common Object Request Broker Architecture (CORBA)

Es un modelo de soporte para la programación distribuida orientada a objetos. Hace posible que los objetos interactúen a través de lenguajes de programación, protocolos de comunicación y plataformas heterogéneas. Este modelo no especifica cómo hacer el soporte, sino qué debe hacer, basado en cinco aspectos de los sistemas distribuidos:

- Interface Definition Language (IDL), que permite la descripción de la interfaz que ofrece un objeto.

- Servicios CORBA, que complementan a los objetos que sirven para la construcción de aplicaciones.
- Interfaces de dominio CORBA, que proveen funcionalidad a usuarios finales en áreas de interés particular.
- General Inter-ORB Protocol, que define los mensajes y el empaquetado de datos que se transmiten entre objetos. Además define su implementación sobre otros protocolos.

3.1.4.2 Remote Method Invocation (RMI)

Posee la misma finalidad que el RPC: invocar de la manera más transparente posible un servicio en una máquina virtual distinta a la que reside el cliente (en la misma máquina física pueden existir varias máquinas virtuales de Java). La diferencia entre estas dos tecnologías radica en que RPC se utiliza en diseños no orientados a objetos, mientras que RMI está soportado por el lenguaje orientado a objetos Java. RMI es un Middleware específico que permite a clientes invocar métodos de objetos como si estuviesen en la misma máquina virtual.

3.1.4.3 Distributed Component Object Model (DCOM)

Al igual que CORBA y RMI, permiten la comunicación de objetos, pero únicamente para las diferentes versiones del sistema operativo Windows. DCOM surge como evolución de Object Linking and Embedding (OLE) y Component Object Model (COM).

3.1.5 Messaging Oriented Middleware (MOM)

Es el Middleware orientado a mensajes; está diseñado para el servicio de mensajes con tecnología asíncrona. Permite el envío de mensajes entre aplicaciones, las aplicaciones sólo ponen y sacan mensajes de las colas, no se conectan. El cliente y el servidor pueden ejecutarse en diferentes tiempos (mensajes asíncronos), por lo que no necesariamente se requiere respuesta.

3.2 La aparición de los Web Services como evolución de los Middleware

Los Web Services son una evolución lógica de los componentes de software y los middleware. Se podría decir que son ambos pero en uno solo.

La aparición de los Web Services se da de la siguiente manera (Figura 3.1):

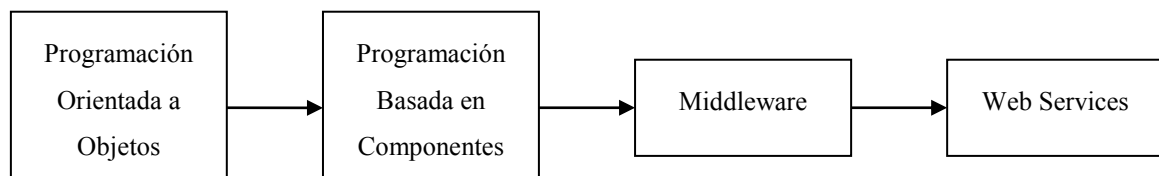


Figura 3.1: La evolución de los Web Services

Decimos que los Web Services son una evolución de los componentes de software porque son una unidad de funcionalidad que expone una interfase, el cual lo describe en base a los mensajes que recibe y genera. La comunicación entre Web Services se hace en términos de mensajes y a través de puertos. La diferencia principal con los componentes es que los Web Services se basan en estándares que permiten la interoperabilidad de las aplicaciones entre organizaciones.

Pero los Web Services tienen una doble naturaleza. Por un lado como ya se mencionó son unidades de funcionalidad y por otro son representados por un conjunto de protocolos (SOAP, WSDL, UDDI). Estos protocolos permiten la interoperabilidad de las aplicaciones y es allí donde tienen cierta similitud con los middleware.

Un middleware es un servicio de un sistema distribuido que tiene una interfase de programación estándar y protocolos. Residen entre las aplicaciones y el sistema operativo junto con la red [Karastoyanova+03]. Los Web Services son considerados como una combinación de protocolos estándar y son accedidos a través de una interfase simple y unificada. Esto implica que los Web Services no son solo su implementación sino también los protocolos e interfases que la tecnología de éstos define. Además la introducción de los Web Services fue con la intención de resolver los problemas de heterogeneidad en sistemas distribuidos, como lo hacen los middleware.

Debido a que los Web Services poseen características de componentes y a la vez de middleware, ha tenido el potencial para convertirse en la principal tecnología para resolver los problemas de integración.

3.3 Comparación de las alternativas de solución al problema dado

A continuación vamos a hacer una comparación entre dos posibles soluciones para el caso de estudio específico presentado en esta tesina. Las alternativas son las siguientes:

- Solución de integración con una arquitectura tradicional cliente/servidor.
- Solución de integración con una arquitectura orientada a servicios.

3.3.1 Características de la solución tradicional cliente/servidor

- Es una arquitectura orientada a la funcionalidad.
- Diseñada para durar.
- Tiene largos ciclos de desarrollo.
- Fuertemente integrada.
- Orientada a datos.

3.3.2 Características de la solución con una arquitectura orientada a servicios

- Esta orientada a procesos.
- Diseñada para el cambio.
- Desarrollo iterativo.
- Ligeramente integrada.
- Orientada a los servicios del negocio.

3.3.3 Parámetros a tomar en cuenta para la elección de la alternativa

Los parámetros que se tomaron en cuenta para seleccionar la solución que mas se adecue a nuestras necesidades son los siguientes:

PR1: Los sistemas que se implementen a futuro deben integrarse de manera fácil: A medida que la organización vaya creciendo la necesidad de desarrollar otros sistemas o adquirirlos se va a hacer presente. La integración de estos nuevos sistemas debe hacerse

de tal manera que no afecte el desempeño del personal y la organización, y el tiempo de adaptación al ambiente organizacional debe ser el menor posible.

PR2: Reutilización de las funcionalidades de las aplicaciones para no volverlas a desarrollar: Este parámetro es importante porque nos va a permitir reducir los costos y tiempos de desarrollo ya que si podemos reutilizar funcionalidades que ya se encuentran implementados en otros sistemas, se reduce la cantidad de código a programar y por lo tanto reducción de tiempos.

PR3: Respuesta rápida al cambio: Las organizaciones cambian constantemente y estos cambios vienen como respuesta a las exigencias del mercado. Cada vez tenemos más competencia y esto nos obliga a generar ventajas competitivas ofreciendo más servicios de calidad. Si las aplicaciones de TI que dan soporte a la organización no se adaptan fácilmente a los cambios que se requieren, la organización no va a poder lograr generar estas ventajas que tanto necesita.

PR4: Bajo coste de implementación: Debido a que la empresa en estudio es relativamente pequeña, los bajos costos en la implementación es un factor importante. Deben utilizarse de preferencia los recursos disponibles.

A continuación se muestra un cuadro comparativo entre las dos arquitecturas y como satisfacen los parámetros definidos anteriormente:

Parámetro	Arquitectura Tradicional Cliente/Servidor	Arquitectura Orientada a Servicios
PR1	Debido a que los sistemas construidos sobre la arquitectura cliente servidor tienen un acoplamiento fuerte, resulta complicado integrar nuevas aplicaciones.	Los servicios que se exponen en una arquitectura orientada a servicios son de bajo acoplamiento lo que permite un ambiente habilitado para la integración con nuevas aplicaciones o aplicaciones ya existentes.
PR2	Una funcionalidad tiene que ser programada cada vez que se necesite en otra aplicación existente.	La arquitectura orientada a servicios permite exponer funcionalidades como servicios y estos pueden ser reutilizados por otras aplicaciones cuando se requieran.
PR3	Debido a que este tipo de aplicaciones con esta arquitectura se han diseñado para durar, hacer algún cambio implica mucha modificación de código, lo que la hace mas compleja.	Debido al bajo acoplamiento que hay entre los servicios, estos pueden ser cambiados sin afectar a aquellos que consuman de él, ya que solo es necesario conocer la interfaz del servicio más no su implementación.
PR4	Los costos serian relativamente bajos, ya que existe toda la plataforma para la construcción de sistemas cliente/servidor en la empresa.	Los costos serían un poco mayores debido a que tiene que implementarse el ambiente para el funcionamiento de servicios Web, pero a largo plazo el hecho de poder reusar funcionalidades disminuye grandemente el costo de construir futuras aplicaciones.

Tabla 3.1: Comparación de las soluciones de acuerdo a los parámetros establecidos

La alternativa factible que elegimos, después de hacer un análisis de los parámetros que debe cumplir para satisfacer los requerimientos de la organización es la alternativa 2: Solución de integración basada en una Arquitectura Orientada a Servicios, debido a que se ajusta más a las necesidades y recursos que posee la empresa en estudio para adoptar este tipo de soluciones.

3.4 Descripción de la Metodología a Utilizar para el desarrollo del caso de estudio

Para el desarrollo del caso de la presente tesina, se ha utilizado una metodología propia basada en criterios obtenidos de la experiencia de trabajo. A continuación se van a nombrar los pasos que se siguieron para diseñar la arquitectura de software basado en SOA:

- Paso 1: Definir los procesos de negocio de la empresa que tienen deficiencias en la integración de los sistemas que lo soportan. Así podemos saber como debe ser el flujo de información entre los sistemas que soportan dicho proceso.

- Paso 2: Luego de que se tienen definidos los procesos sobre los cuales se va a hacer la integración, se realiza el análisis de requerimientos, tanto funcionales como no funcionales.
- Paso 3: Escoger que patrones se van a utilizar que satisfagan los requerimientos funcionales y no funcionales definidos en la fase previa. Para la selección de estos patrones es muy importante tener en cuenta que la solución está basada en SOA para la integración, así que el uso de éstos debe habilitar las principales características de una solución basada en servicios de bajo acoplamiento.
- Paso 4: Luego de definirse los patrones, debe procederse a representar la arquitectura en base a vistas. Para esto debe seleccionarse un Framework. En nuestro caso el Framework a utilizar es el de Vistas “4+1”.
- Paso 5: Deben definirse los estándares y la tecnología a usarse para el proyecto de integración.
- Paso 5: Se realiza el diagrama de casos de uso donde se visualizan los módulos o subsistemas que conforman la solución.
- Paso 6: Se procede a la descripción de la Vista Lógica que comprende los casos de uso de cada módulo y el diagrama de secuencias correspondiente a cada caso de uso.
- Paso 7: Se define la Vista de Procesos.
- Paso 8: Se define la vista de implementación donde se muestran los componentes y de que manera interactúan.
- Paso 9: Se define la vista de despliegue que es el mapeado de esos componentes en el hardware.
- Paso 10: Por último se definen los atributos de calidad a ser tomados en cuenta en la implementación de la solución.

A continuación se muestra un cuadro resumen:

Actividad	Entregable
Definir los procesos de negocio que se tomarán en cuenta en la solución.	Modelo de Procesos de Negocio
Análisis de requerimientos	Requerimientos funcionales y no funcionales.
Selección de patrones para la solución SOA.	Listado de patrones a utilizar y descripción de su implementación.
Selección del Framework a utilizar para definir la arquitectura.	Nombre y descripción del Framework a utilizar.
Selección de estándares y la tecnología para la solución.	Definición de estándares y su forma de uso. Definición de la tecnología a utilizar.
Desarrollo de la vista de casos de uso.	Diagramas de caso de uso y especificaciones de caso de uso.
Desarrollo de la vista Lógica.	Diagramas de clase y de secuencia.
Desarrollo de la vista de procesos.	Diagrama de la arquitectura de procesos.
Desarrollo de la vista de implementación.	Modelo de Implementación.
Desarrollo de la vista de despliegue	Diagrama de despliegue
Requerimientos de atributos de calidad	Listado de atributos de calidad

Tabla 3.2: Resumen de la Metodología a utilizar

Capítulo 4: Desarrollo del Caso

4.1 La Asociación Los Profesionales Aéreos – Professional Air

Professional Air es una institución que se dedica a formar profesionales de la Aviación Civil. Las carreras que ofrece son: Aviación Comercial, Flight Hostess, Counter Profesional, Piloto Privado, Piloto Comercial y Despachador de Vuelo.

4.1.1 Visión

Ser a mediano plazo la mejor Institución Educativa de Aviación Comercial, reconocida mundialmente por instituciones y personas, debido a la calidad de enseñanza y práctica de nuestros valores.

4.1.2 Misión

Brindar servicios educativos de calidad, con el uso adecuado de nuestra alta capacidad instalada, personal profesional calificada y con sentido humanista; que garantice la formación técnica del estudiante de Aviación Comercial, que le permita ocupar un puesto de trabajo o administrar su propia empresa.

4.1.3 Objetivos Estratégicos

OBJETIVO DE GESTIÓN INSTITUCIONAL

En “PROFESSIONAL AIR”, consideramos como uno de nuestros objetivos fundamentales de Institución Educativa elevar permanentemente los niveles académicos y la calidad de nuestros servicios en las diferentes especialidades que brindamos a los jóvenes estudiantes, fomentando la actualización, capacitación y motivación de todo el personal docente y administrativo, que les permita asumir con espíritu de amor y servicio su compromiso de orientar y educar.

OBJETIVO DE GESTIÓN PEDAGÓGICA

El CEOP ”PROFESSIONAL AIR”, se orienta a desarrollar una formación integral de los alumnos, con práctica de valores y de prevención, una formación académica, humana, científica y cristiana acordes a nuestra realidad y los grandes avances dentro del sistema globalizada, que impulse el desarrollo de las capacidades, habilidades y

destrezas en una orientación hacia la iniciativa, creatividad, el emprendimiento, la participación y trabajo, que le permita al alumno adecuarse al desarrollo productivo del país y enfrentarse a la vida con dignidad e identidad nacional.

OBJETIVO DE GESTIÓN ADMINISTRATIVA.

El CEOP "PROFESSIONAL AIR", considera objetivo, buscar permanentemente la integración y el cambio de todos los integrantes de la comunidad Educativa mediante el conocimiento y solución de los problemas y dificultades practicado en el trabajo comprometido, participativo y democrático y mejorado los niveles de comunicación y coordinación.

4.1.4 Estructura Orgánica

La estructura orgánica de Professional Air esta constituida como se muestra en la Figura 4.1, de un conjunto de unidades ordenadas e interrelacionadas, destinadas a cumplir funciones establecidas.

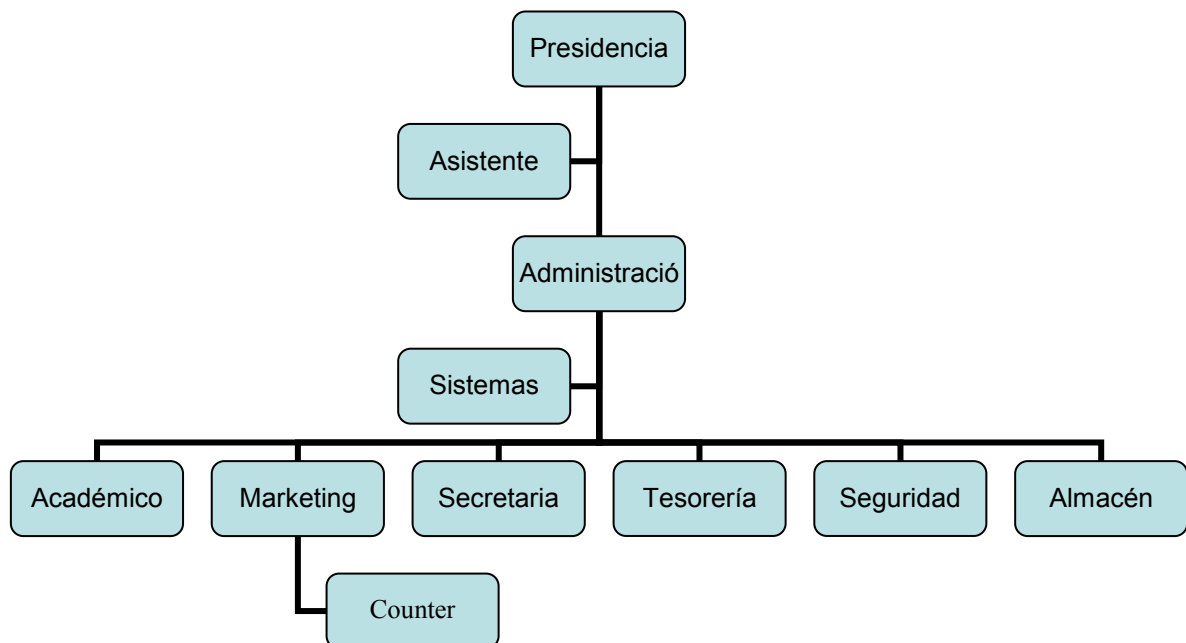


Figura 4.1: Organigrama de Professional Air

4.2 Problemática Actual

Professional Air cuenta con una serie de aplicativos para el soporte de su carga operativa. A continuación en la tabla 1.1 se nombran los sistemas existentes y sus características tecnológicas:

Sistema	Descripción	Lenguaje de Programación	Base de Datos	Plataforma
Sistema de Gestión Académica	Se encarga de dar soporte a todas las labores del Departamento Académico.	Visual Basic 6	SQL Server 2000 (BDAcadémico)	Windows
Sistema de Matrícula	Es utilizado por el área de Counter para la matrícula de los interesados.	Visual Basic 6	SQL Server 2000 (BDAcadémico)	Windows
Sistema de Contabilidad	Se encarga de dar soporte a todas las labores del Departamento de Contabilidad.	Visual FoxPro	FoxPro (BDContable)	Windows
Sistema de Tesorería	Se encarga de dar soporte a todas las labores del área de tesorería.	Visual Basic 6	SQL Server 2000 (BDTesoro)	Windows
Sistema de Control de Asistencia	Se encarga del registro de asistencia del personal administrativo y docente, del registro de visitas, salidas a comisiones y permisos del personal.	PHP	MySQL	Linux

Tabla 4.1: Sistemas de información en Professional Air

Estos sistemas desarrollados en su mayoría bajo una arquitectura cliente/servidor sobre plataforma Microsoft trabaja cada uno con su propia lógica de negocio y su propia base de datos. Cada sistema es una isla de información a excepción del sistema de Matrícula y Académico que comparten la misma base de datos.

La necesidad de integración de las aplicaciones es un factor clave para poder solucionar los problemas operacionales que se originan debido a la carencia del flujo de información entre los sistemas de la empresa y también van a dar paso a la creación de nuevos servicios, como por ejemplo una intranet para los alumnos donde puedan consultar sus notas, horarios, pagos, etc., y la implementación de e-learning como proyecto futuro.

4.3 Diseño de la Arquitectura de Software para la Integración de los sistemas

4.3.1 Procesos de negocio

Para la presente tesina se han analizado dos procesos de negocio en la empresa de estudio, los cuales tienen un alto grado de importancia debido a que la mayoría de problemas surgidos son a causa de la ineficiente integración de las tareas correspondientes a dichos procesos.

PR1: Proceso de matrícula (Figura 4.2)

Este proceso se realiza cuando una persona interesada se matricula o reserva su matricula en una de las carreras ofrecidas por la institución.

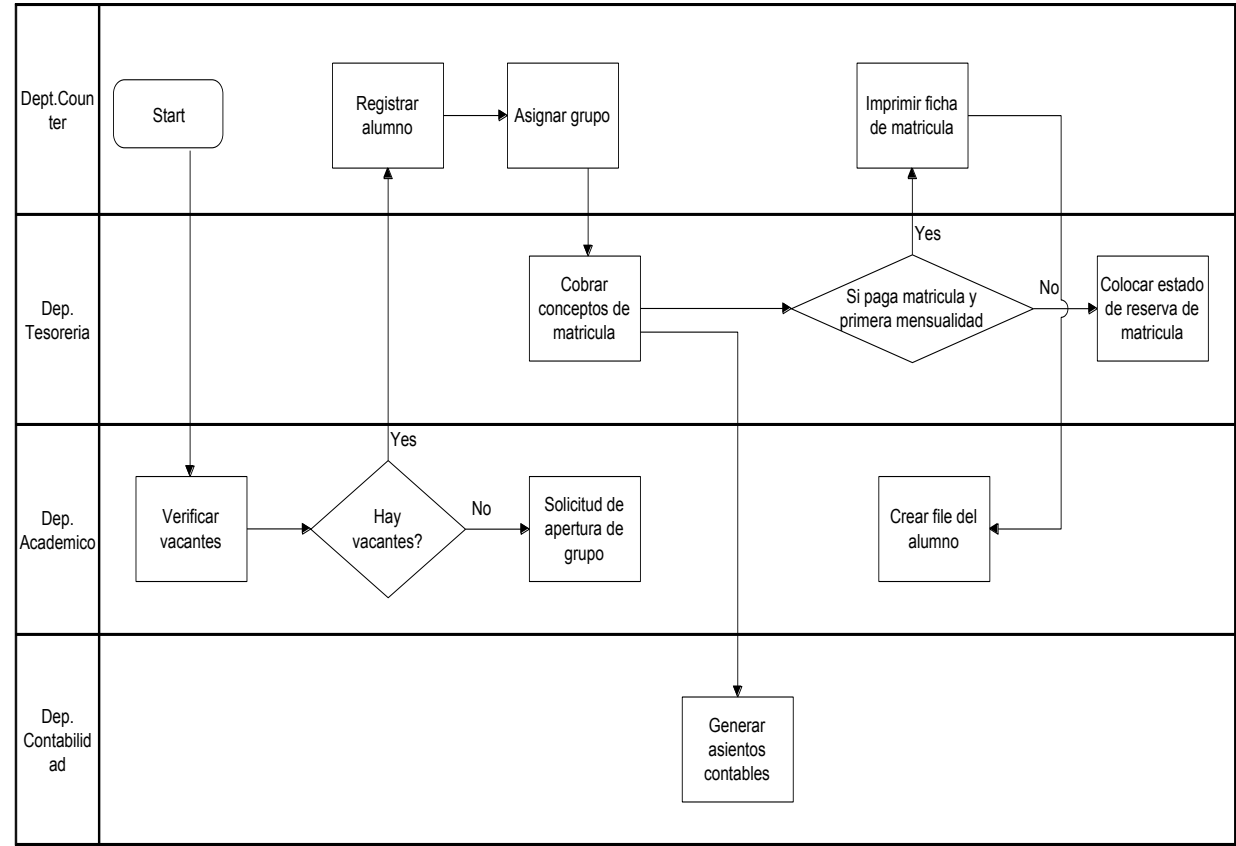


Figura 4.2: Proceso de Matrícula

Este proceso inicia cuando un interesado desea matricularse. El interesado se dirige al Departamento de Counter para ser registrado en el sistema de Matrícula. Previo al registro en el sistema, la persona de Counter debe verificar si existen vacantes para la carrera y el horario que desea el interesado. En caso no existan vacantes, se procede a hacer una solicitud para la apertura de un nuevo grupo. Si aun hay vacantes disponibles se procede con el registro de los datos personales del interesado, los datos de su apoderado y los datos de su matricula. Luego de esto el interesado pasa al Departamento de Tesorería a realizar los pagos correspondientes. Una vez que el interesado ha realizado los pagos, el Departamento de Tesorería genera las boletas, las cuales pasan al Departamento de Contabilidad para generar el asiento contable respectivo. Si el interesado ha pagado los conceptos de matrícula y primera mensualidad, es considerado ya como alumno matriculado y pasa nuevamente al área de Counter donde su ficha es impresa y firmada por él y una copia se deriva al Departamento Académico para crearle su file. Si no ha efectuado ambos pagos, se le considera como reserva de matrícula.

PR2: Proceso de pago de planillas (Figura 4.3)

Este proceso se realiza días antes de finalizar el mes, en el cual se calculan los descuentos por faltas y tardanzas del personal, se generan las planillas y previa aprobación se procede con el pago de las mismas.

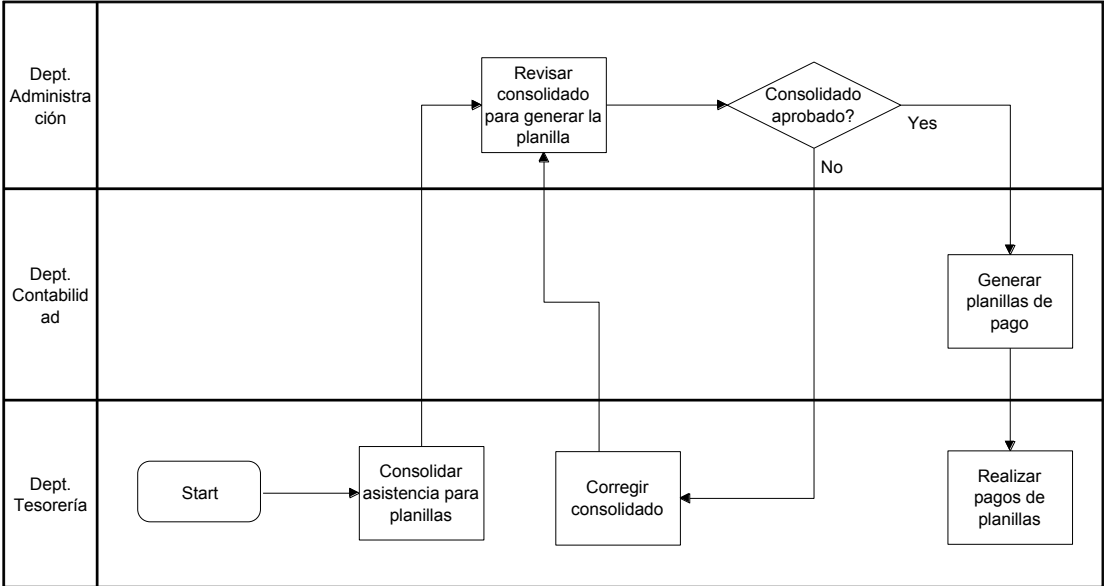


Figura 4.3: Proceso de pago de planillas

Este proceso se inicia cuando el Departamento de Tesorería realiza los consolidados de la asistencia de todo el personal tanto docentes como administrativos para realizar las planillas de pago de fin de mes. Una vez que se han realizado los consolidados, se envían al Departamento de Administración para que haga las revisiones y la aprobación. En caso exista algún error, el consolidado retorna al Departamento de Tesorería para su corrección. Una vez que la Administración aprueba el consolidado, pasa al Departamento de Contabilidad donde se generan las planillas. Luego de esto se procede al pago de ellas en el Departamento de Tesorería.

La solución planteada se enfoca en la integración de las actividades de los procesos descritos anteriormente. Los sistemas existentes cumplen con las tareas pero de manera aislada. Lo que se quiere es que las aplicaciones se comuniquen para el correcto flujo del proceso.

4.3.2 Análisis de Requerimientos

4.3.2.1 Requerimientos Funcionales

RF1: Las planillas generadas en el sistema de Contabilidad para el pago del personal administrativo y docente, debe tomar los datos directamente del sistema de asistencia para calcular los descuentos por faltas y tardanzas de manera automática.

RF2: La programación de horarios de los docentes debe estar centralizada en el Sistema Académico y debe reflejarse automáticamente en el Sistema de Asistencia.

RF3: Se debe considerar que un alumno es matriculado luego de la realización de los pagos correspondientes en el sistema de Tesorería.

RF4: Debe eliminarse la redundancia de información que existe en las diferentes bases de datos de los sistemas.

RF5: La gestión de usuarios y del personal debe ser manejada en un único sistema.

RF6: El sistema de tesorería al recibir algún pago de algún alumno o al hacerle una nota de crédito, debe generar automáticamente el asiento contable en el sistema de contabilidad.

RF7: Exponer la funcionalidad de las aplicaciones existentes como Web Services para permitir la integración con el resto de aplicaciones.

4.3.2.2 Requerimientos no funcionales

RN1: Asegurar la disponibilidad de los servicios.

RN2: Asegurar la integridad de los mensajes entre los servicios.

RN3: Asegurar la confidencialidad de los mensajes para que sólo las personas autorizadas puedan acceder a la información.

RN4: Disponer de una alta usabilidad para disminuir la necesidad de capacitación de los usuarios.

RN5: Debe utilizarse los recursos humanos disponibles en la empresa.

4.3.3 Patrones y Framework de Arquitectura a utilizarse

La arquitectura sigue el framework “4+1”, el cual presenta cuatro vistas para la arquitectura en conjunto con los escenarios. Más adelante se define cada una de estas vistas para la representación de la arquitectura del caso de estudio dado.

Los patrones de arquitectura a usarse en el diseño son:

- Patrón N-capas: Se ha definido 4 capas para el desarrollo de la solución: la capa de presentación, capa de servicios, lógica de negocios y la capa de entidades. La capa de presentación es opcional para el acceso a servicios desde otras aplicaciones ya existentes, las cuales se comunicarán directamente con el proveedor del servicio en la capa de servicios.
- Patrón de Mensajes Request/Response: Se requiere de este patrón debido a que los sistemas consumidores de servicios necesitan una respuesta inmediata del proveedor del servicio. El protocolo de comunicación utilizado es el HTTP.
- Patrón Broker de mensajes: Un broker separa los sistemas origen de los sistemas destino asumiendo la responsabilidad de coordinar la comunicación entre los puntos finales. Para que la solución planteada sea SOA y se cumpla con la característica de acoplamiento débil, es que se va a usar el broker para el ruteo y el registro de los Web Services.

A continuación se detallan cada una de las capas, indicándose las características correspondientes:

4.3.3.1 Capa de Presentación

La capa de presentación es la encargada de manejar la interacción con los usuarios. La capa contiene componentes de interfaz que son formularios Web de Asp.Net. Los datos mostrados en esta capa son el resultado de la invocación de componentes de la lógica de negocios o de la llamada a Servicios Web.

Se van a definir capas de presentación para los módulos de Gestión de Usuarios y Gestión de Personal, los cuales se van a manejar de manera centralizada en una única aplicación Web.

4.3.3.2 Capa de servicios

En esta capa se van a definir los servicios que se desean exponer para ser consumidos por las diferentes aplicaciones. Estos servicios son los que tienen acceso a los componentes en la capa de negocio. Todos los servicios Web van a ser desarrollados en .Net.

A continuación se van a listar los servicios Web que se implementarán para la integración de los sistemas en Professional Air:

Sistema que provee el servicio	Nombre del servicio	Descripción	Sistemas que lo consumen
Sistema de Asistencia	WS_Asiistencia	Hace la consolidación de la asistencia del personal para la generación de planillas. Métodos expuestos: Consolidar_asistencia(fechainicio,fechafin)	Sistema de Contabilidad
Sistema de Contabilidad	WS_Asiento	Genera automáticamente el asiento contable ya sea para pagos de los alumnos o devoluciones a los mismos. Métodos expuestos: Generar_asiento(tipo,conceptos)	Sistema de Tesorería
Sistema Académico	WS_Horario	Devuelve los horarios de un docente para el marcado de su asistencia. Métodos expuestos:	Sistema de Asistencia

		Consultar_horario(cod_docente)	
Integrador	WS_Alumno	Se encarga de toda la gestión de alumnos. Métodos a exponer: Ingresar_alumno(datos_alumno, datos_apoderado, datos_matricula) Cambiar_estado(tipo, estado, cod_matricula, cod_alumno) Modificar_alumno(campos, valores) Modificar_apoderado(campos, valores)	Sistema de Matricula, Sistema Académico, Sistema de Contabilidad, Sistema de Tesorería
Integrador	WS_Carrera	Devuelve los datos de las carreras solicitadas. Métodos a exponer: Obtener_carreraporcodigo(codigo) Listar_carreras()	Sistema Académico, Sistema de Matricula, Sistema de Tesorería
Integrador	WS_Personal	Devuelve los datos y listados del personal solicitado. Métodos a exponer: Listar_personal(tipo) Consultar_personal(tipo, cod_personal) Listarpersonalsinhorario() Listar_tipo_planillas()	Sistema de Asistencia, Sistema de Contabilidad, Sistema Académico, Sistema de Tesorería
Integrador	WS_Usuario	Hace la validación de usuarios para cada sistema. Métodos a exponer: Validar_usuario(usuario, pass, sistema)	Sistema de Matricula, Sistema de Tesorería, Sistema de Contabilidad, Sistema de Asistencia
Integrador	WS Modalidad	Devuelve los datos y el listado de las modalidades solicitadas. Métodos a exponer: Consultar_modalidad(codigo) Listar_modalidadesvigentes()	Sistema de Matricula, Sistema de Tesorería

Tabla 4.2: Listado de servicios a exponer

4.3.3.3 Capa de Negocio

En esta capa van a ir definidos los componentes principales del negocio que son utilizados por los servicios web. Se implementarán como Class Libraries en .Net.

4.3.3.4 Capa de Datos o Entidades

Aquí se definen los componentes que tienen acceso a las diferentes fuentes de datos usados por cada aplicación. Lo que se ha propuesto es eliminar la redundancia de datos en las diferentes bases de datos y tener una sola base de datos común donde se centralicen los datos compartidos por todas las aplicaciones. Esta base de datos tiene las siguientes tablas:

- Alumno, Matricula, Carrera, Modalidad, Apoderado, Distrito, Personal, TipoPersonal, TipoPlanilla, Area, Usuario, TipoUsuario, Sistema, Opción, Acceso.

4.3.3.5 El Broker de Mensajes

Para la solución propuesta se va utilizar Microsoft BizTalk Server 2006 R2. Aquí es donde se van a registrar los servicios para que los consumidores se enlacen con los proveedores sin necesidad de conocer su ubicación física.

Para un mayor entendimiento de cómo funciona el broker de mensajes aplicado a nuestra solución, mostramos la siguiente figura:

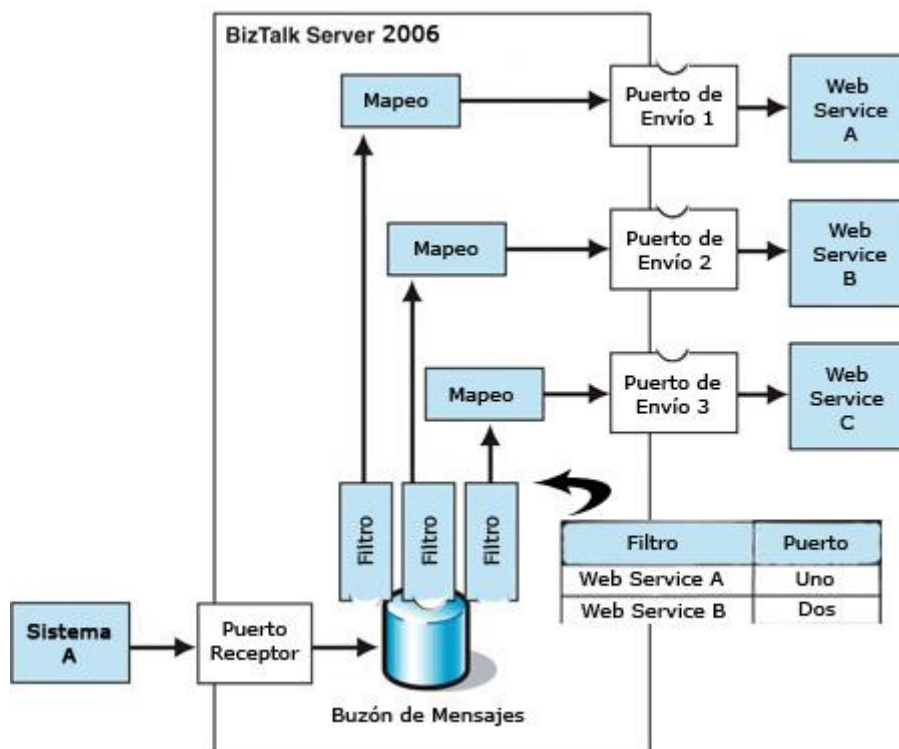


Figura 4.4: Descripción de la funcionalidad de BizTalk Server

Como se muestra en la figura 4.4, el sistema que consume el servicio envía el mensaje a un puerto receptor en el BizTalk Server y es almacenado en un buzón. Dependiendo de que servicio desee consumir, es que pasa por un filtro donde se le asigna un número de puerto por el cual el mensaje es transmitido al servicio proveedor.

4.3.4 Uso de estándares y tecnología

4.3.4.1 Estándares

Para el modelo de procesos de negocio se debe usar la notación BPMN, definida como estándar en la empresa.

Se debe utilizar UML para la documentación propia del sistema entre los miembros del equipo de desarrollo como con los clientes y usuarios.

Las nuevas interfaces deben estar orientadas a la Web y deberá poder ser vista con cualquiera de los browsers que existen actualmente.

Para la interoperabilidad entre los sistemas existentes se deben usar Web Services. Los formatos de mensajes deberán estar en XML, el protocolo para el intercambio de mensajes deberá ser SOAP y el protocolo de transporte será HTTP.

4.3.4.2 Tecnología

El desarrollo de los servicios a exponer debe estar realizado en Microsoft .Net, basándose en las tecnologías involucradas en la plataforma como son ADO .Net y ASP .Net. El lenguaje de programación será VB .Net.

Para el servicio expuesto por el Sistema de Asistencia se utilizará PHP.

El broker de mensajes a utilizar será el Microsoft BizTalk Server 2006 como se mencionó anteriormente.

Los servicios realizados en .Net serán deployados sobre el BizTalk Server, que cuenta con soporte para Web Services, que se encuentra configurado en un servidor Microsoft Windows Server 2003. El servicio realizado en PHP se deployará en un servidor Web Apache en plataforma Linux.

4.3.5 Vista de Casos de Uso

Hemos separado los casos de uso en paquetes, como se ve en la Figura 4.4, que representan los módulos y servicios que se van a exponer para integrar los diferentes sistemas.

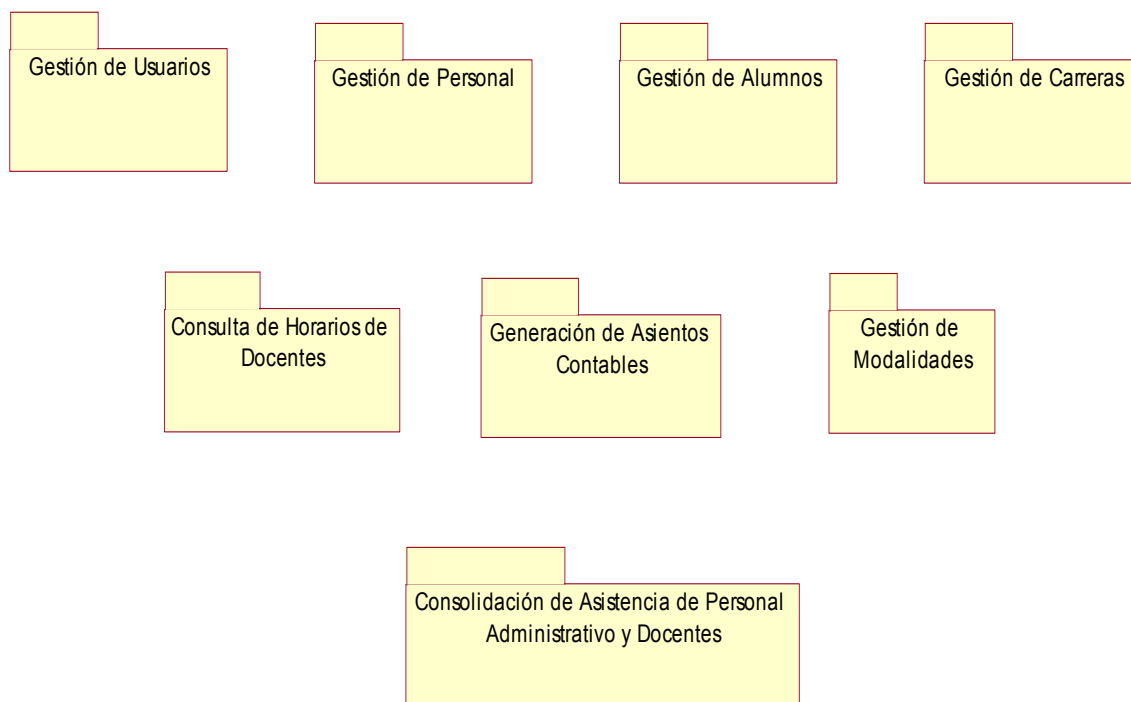


Figura 4.5: Diagrama de casos de uso

4.3.5.1 Gestión de Usuarios

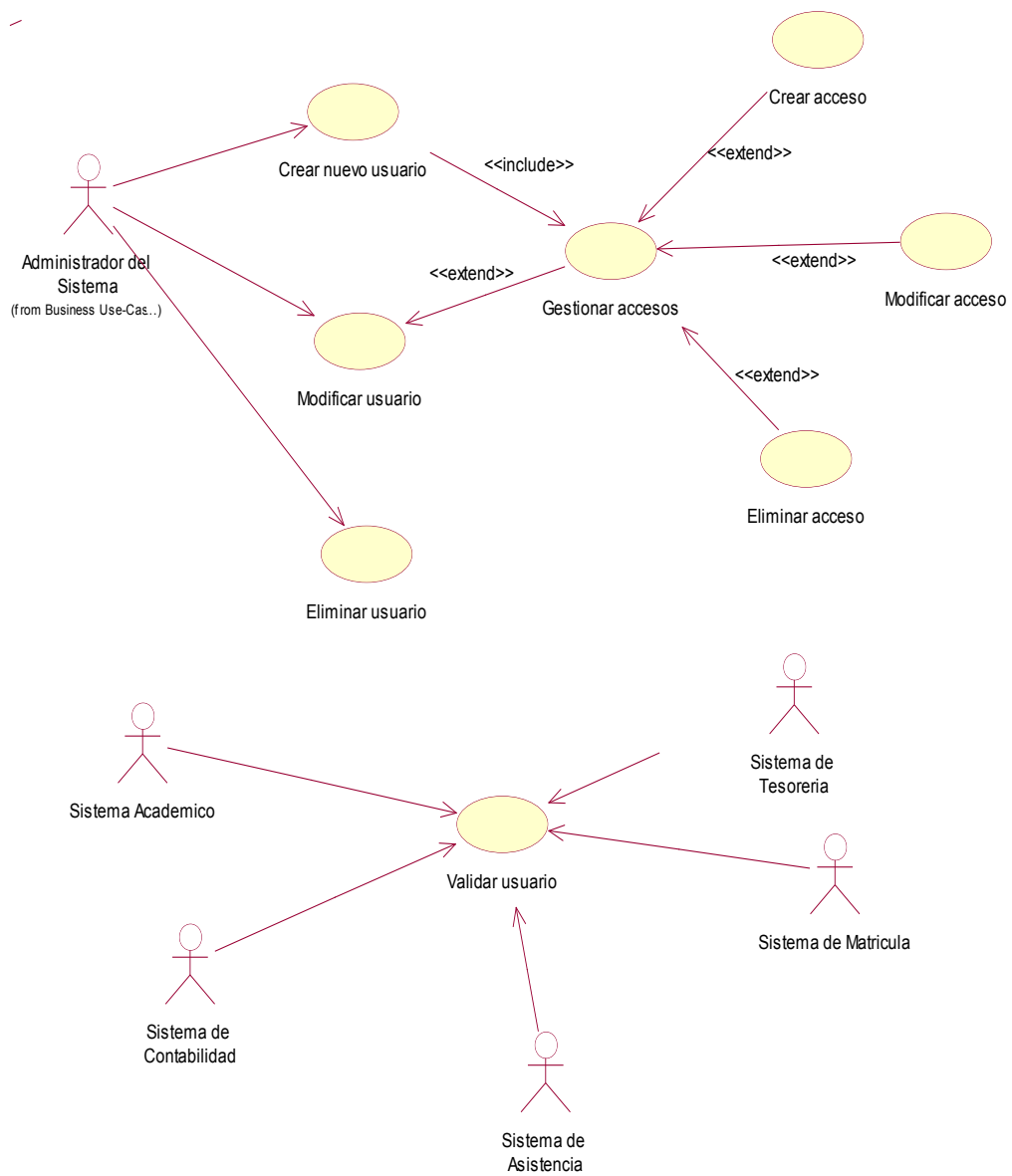


Figura 4.6: Casos de Uso del modulo Gestión de Usuarios

4.3.5.2 Gestión de Personal

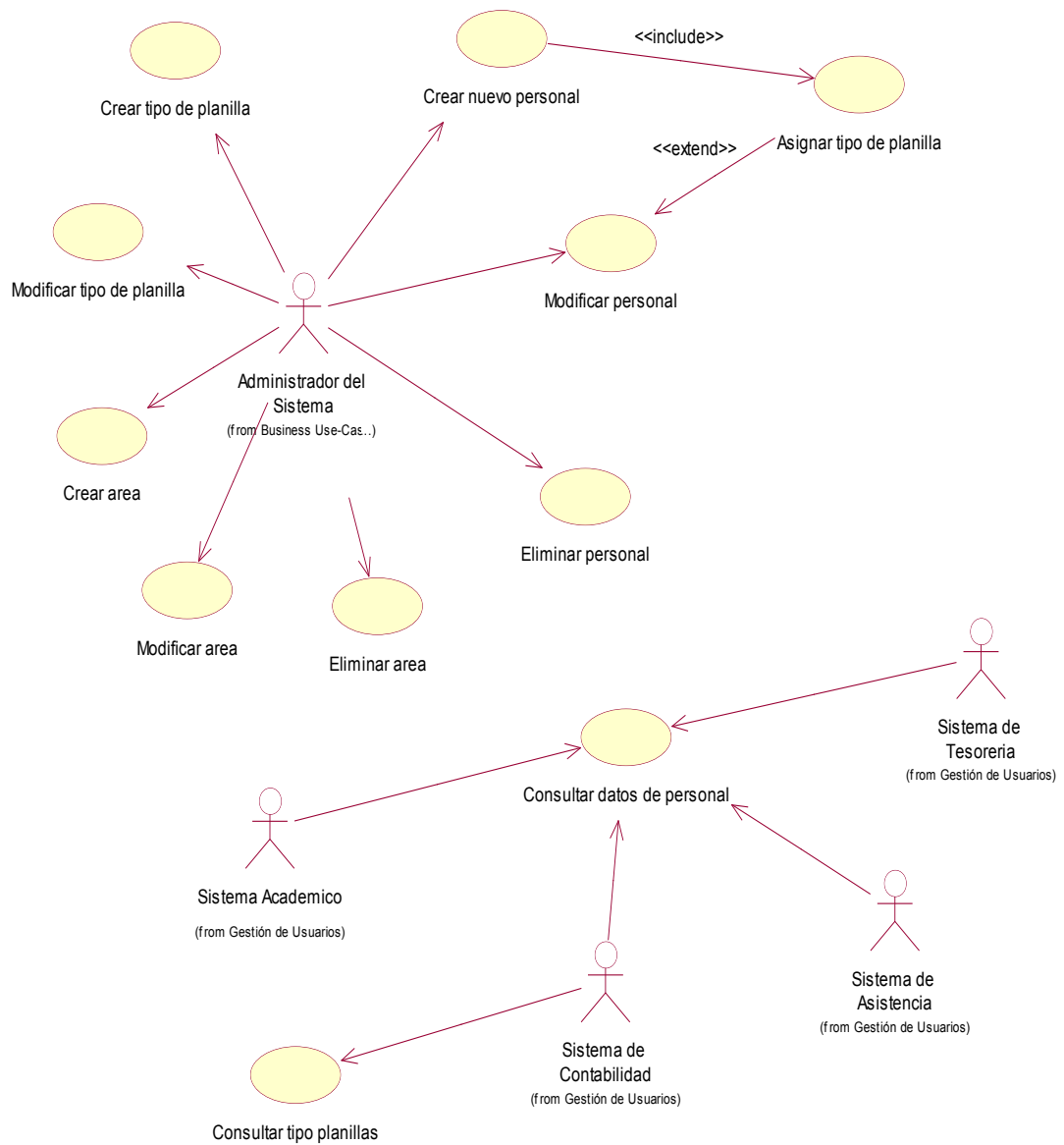


Figura 4.7: Casos de Uso del módulo de Gestión de Personal

4.3.5.3 Gestión de Alumnos

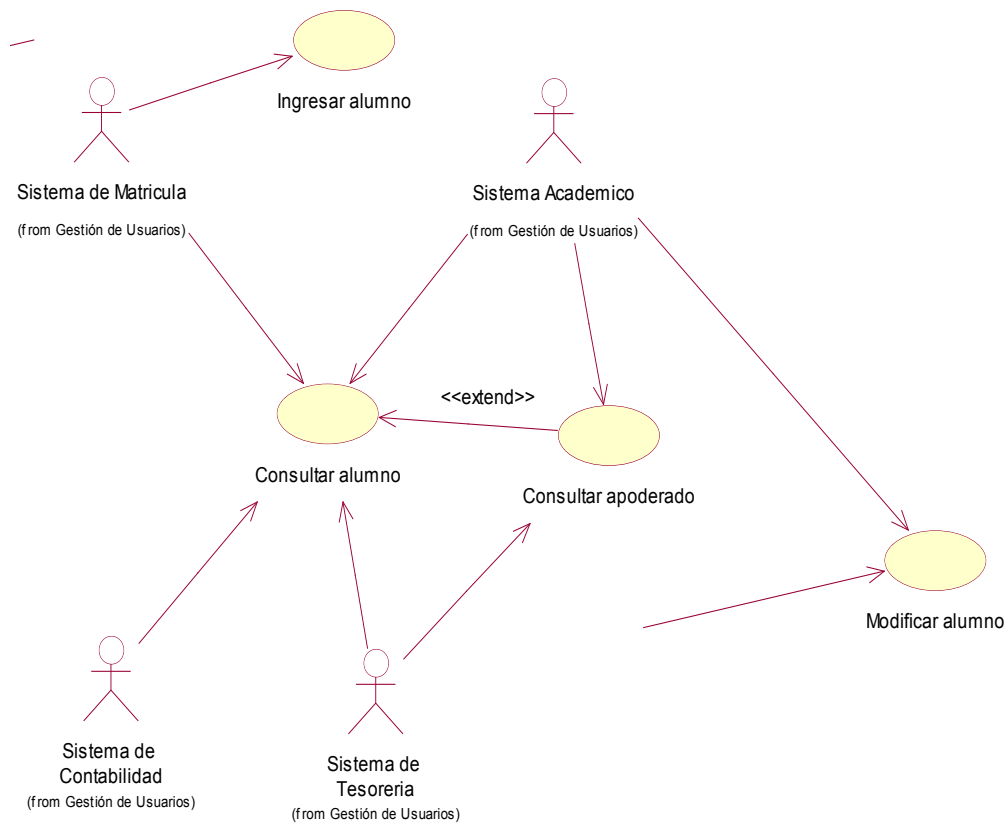


Figura 4.8: Casos de Uso del módulo de Gestión de Alumnos

4.3.5.4 Consulta de horarios de docentes



Figura 4.9: Casos de Uso del módulo de Consulta de Horarios de Docentes

4.3.5.5 Generación de Asientos Contables

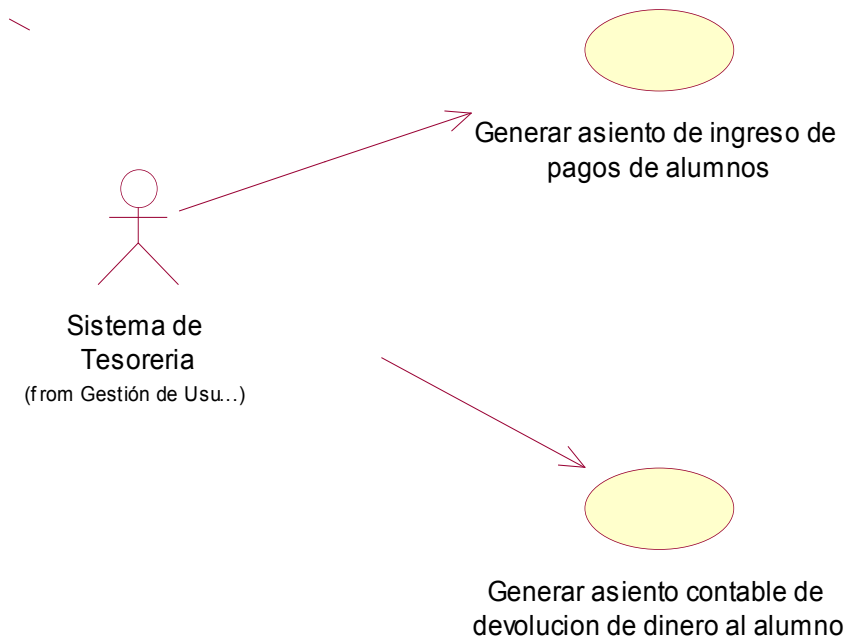


Figura 4.10: Casos de Uso del módulo de Generación de Asientos Contables

4.3.5.6 Gestión de Carreras

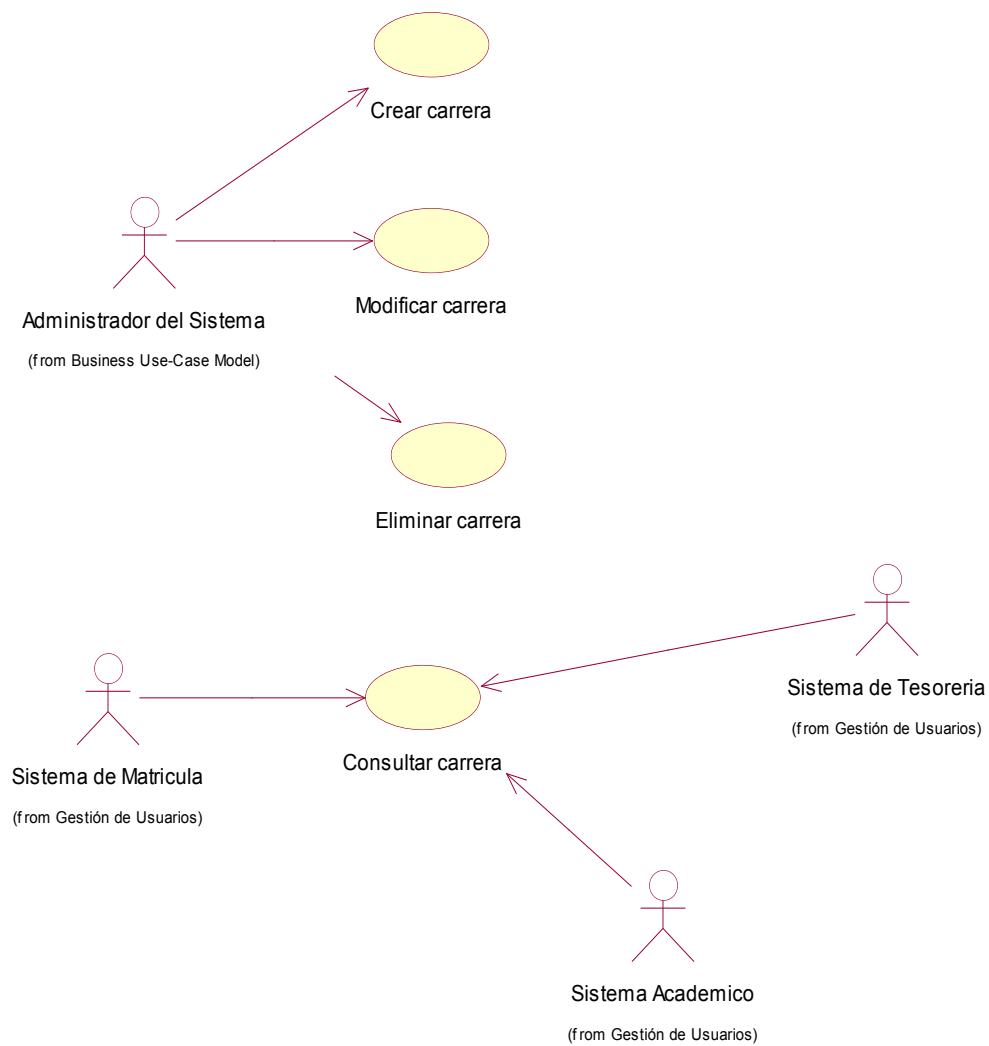


Figura 4.11: Casos de Uso del módulo de Gestión de Carreras

4.3.5.7 Consolidación de Asistencia de Personal y Docentes

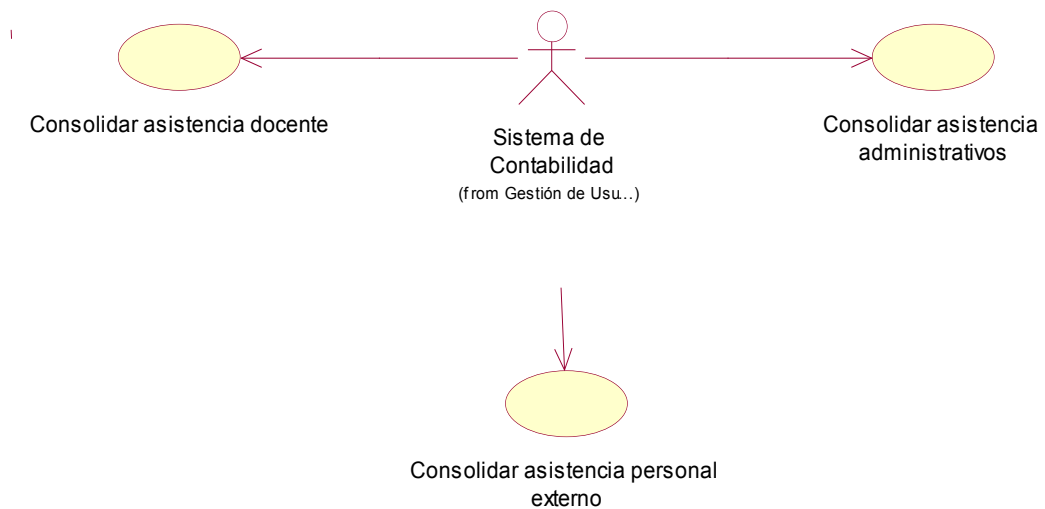


Figura 4.12: Casos de Uso del módulo de Consolidación de Asistencia de Personal y Docentes

4.3.5.8 Gestión de Modalidades de Matrícula

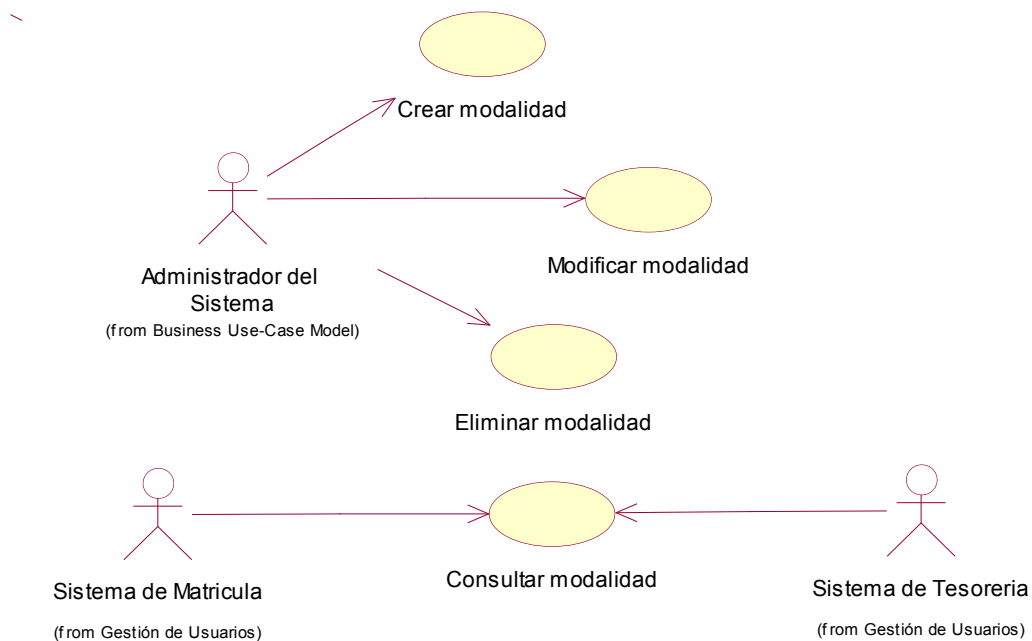


Figura 4.13: Casos de Uso del módulo de Gestión de Modalidades de Matrícula

4.3.6 Vista Lógica

En esta vista es donde los requerimientos funcionales o casos de uso se ven realizados en clases y su organización en paquetes. Hemos separado los diagramas por los módulos que se definieron al presentar los casos de uso.

4.3.6.1 Módulo Gestión de Usuarios

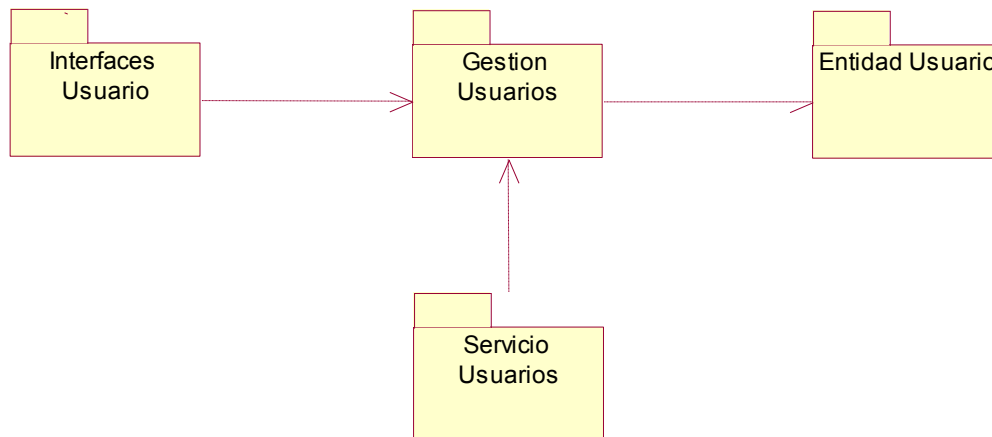


Figura 4.14: Diagrama de paquetes del módulo de gestión de usuarios

Se han definido las 4 capas en este módulo, las cuales se detallan a continuación:

- Interfaces Usuario:



Figura 4.15: Diagrama de Clases interfaz del módulo de gestión de usuarios

- Servicio Usuario

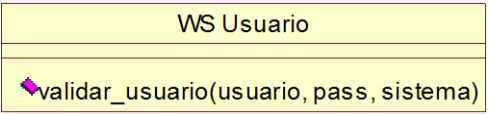


Figura 4.16: Definición del Servicio de Gestión de Usuarios

- Gestión Usuario

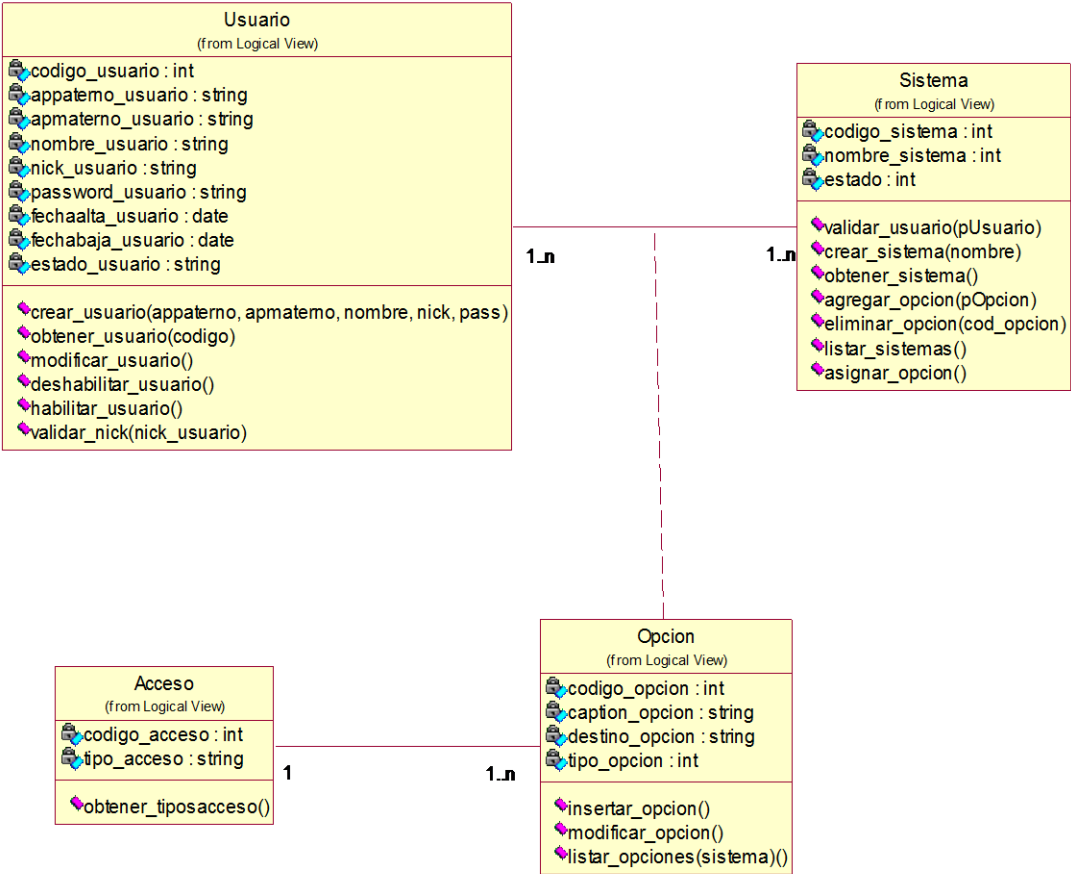


Figura 4.17: Diagrama de clases de la capa de negocio del módulo de gestión de usuarios

- Entidad Usuario

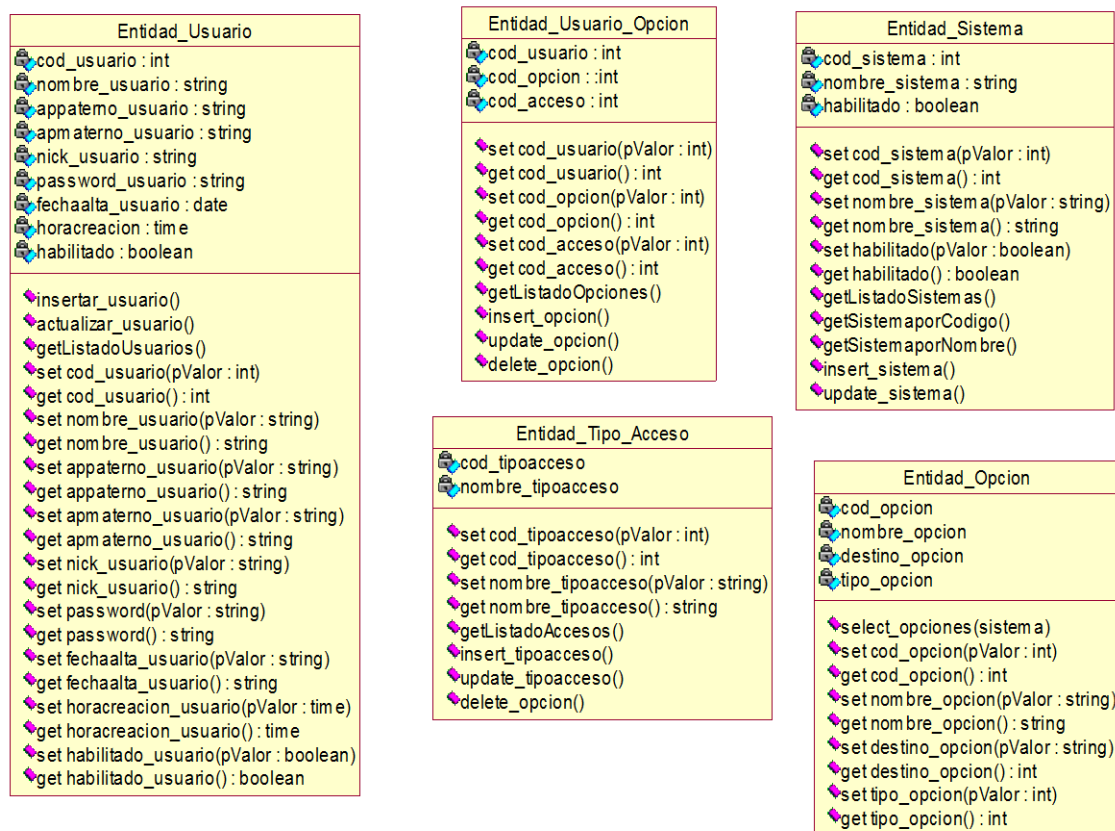


Figura 4.18: Diagrama de Clases Entidad del módulo de gestión de usuarios

4.3.6.2 Módulo de Gestión de Personal

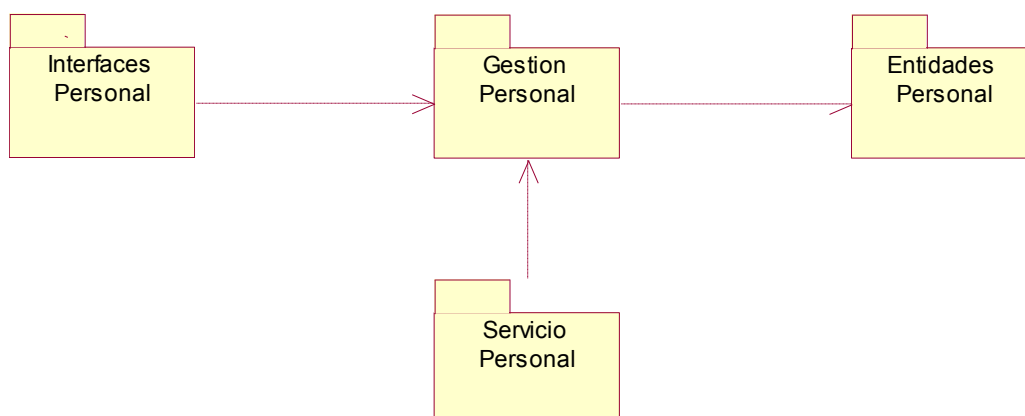


Figura 4.19: Diagrama de paquetes del módulo de gestión de personal

Se han definido las 4 capas en este módulo, las cuales se detallan a continuación:

- Interfaces Personal

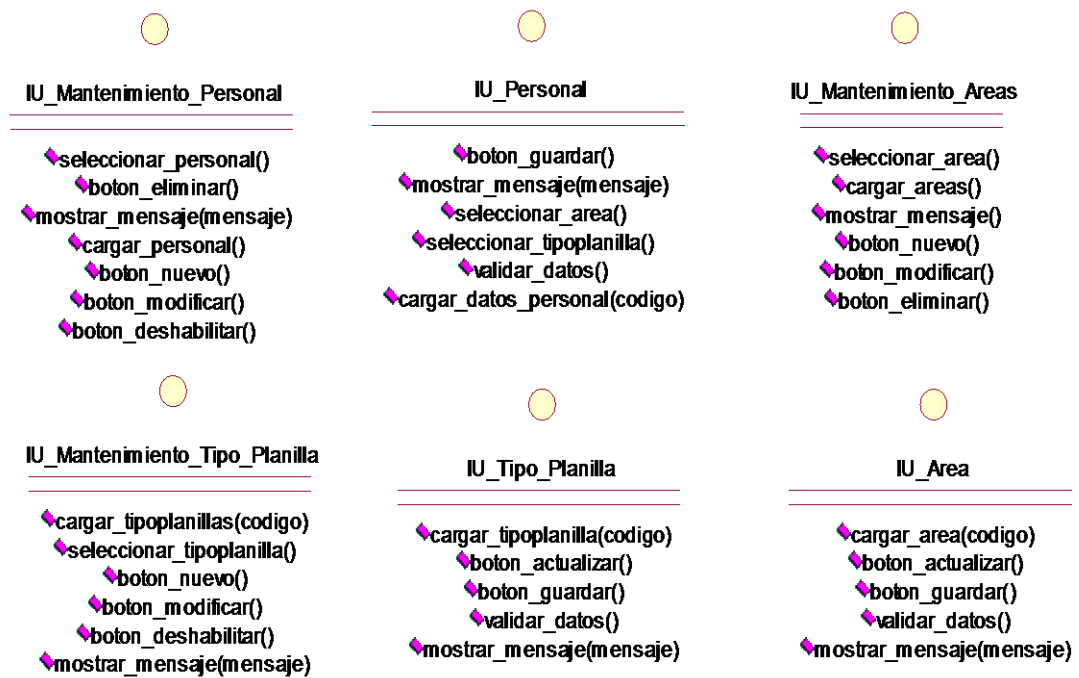


Figura 4.20: Diagrama de Clases interfaz del módulo de gestión de personal

- Servicio Personal

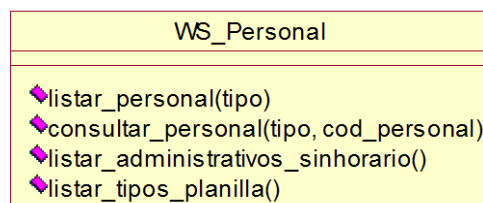


Figura 4.21: Definición del Servicio de Gestión de Personal

- Gestión Personal

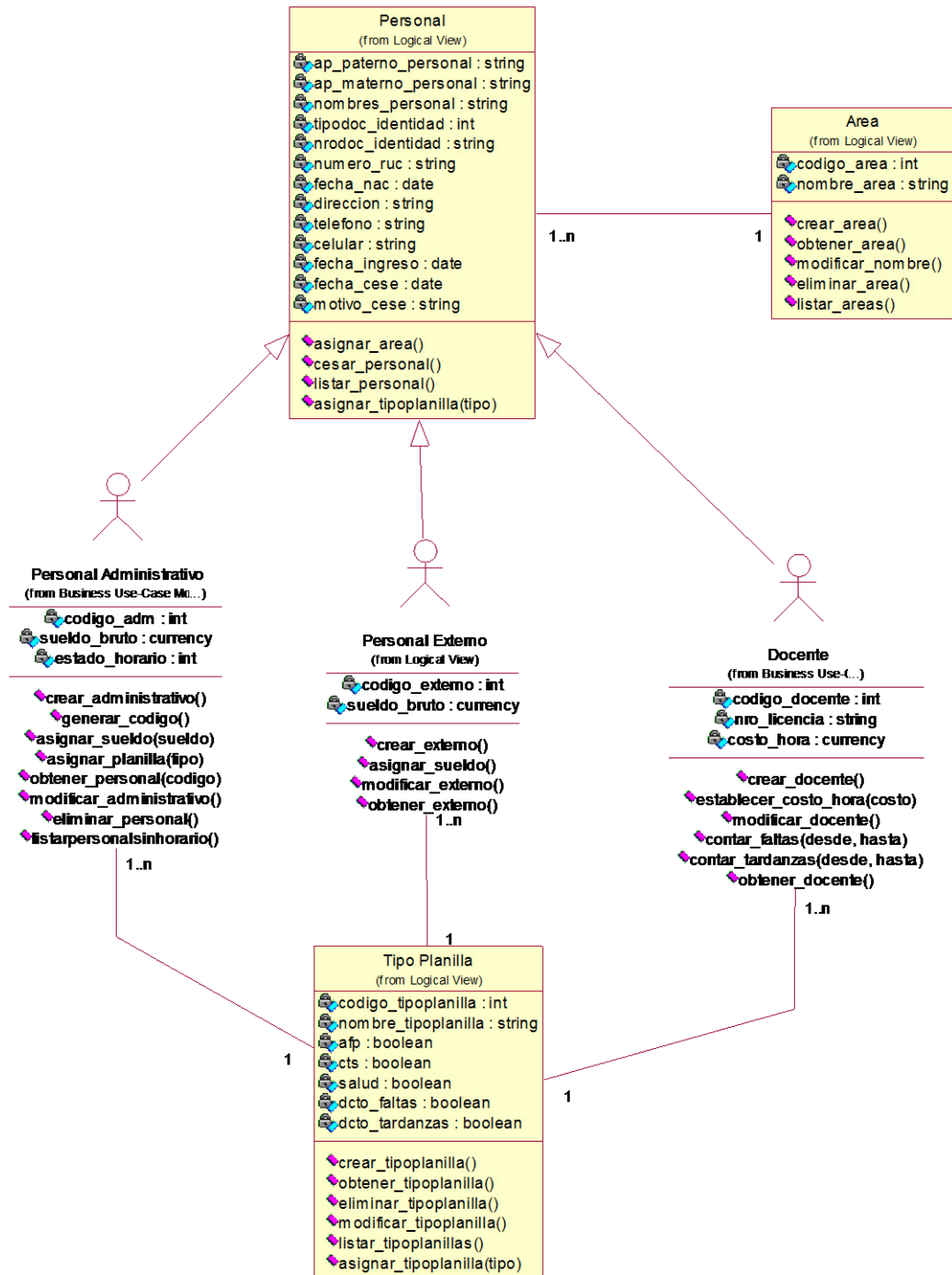


Figura 4.22: Diagrama de clases de la capa de negocios del módulo gestión de personal

- Entidades Personal



Figura 4.23: Diagrama de Clases entidad del módulo de gestión de personal

A continuación se muestra los diagramas de secuencia para los métodos definidos en el servicio Web que se va a exponer:

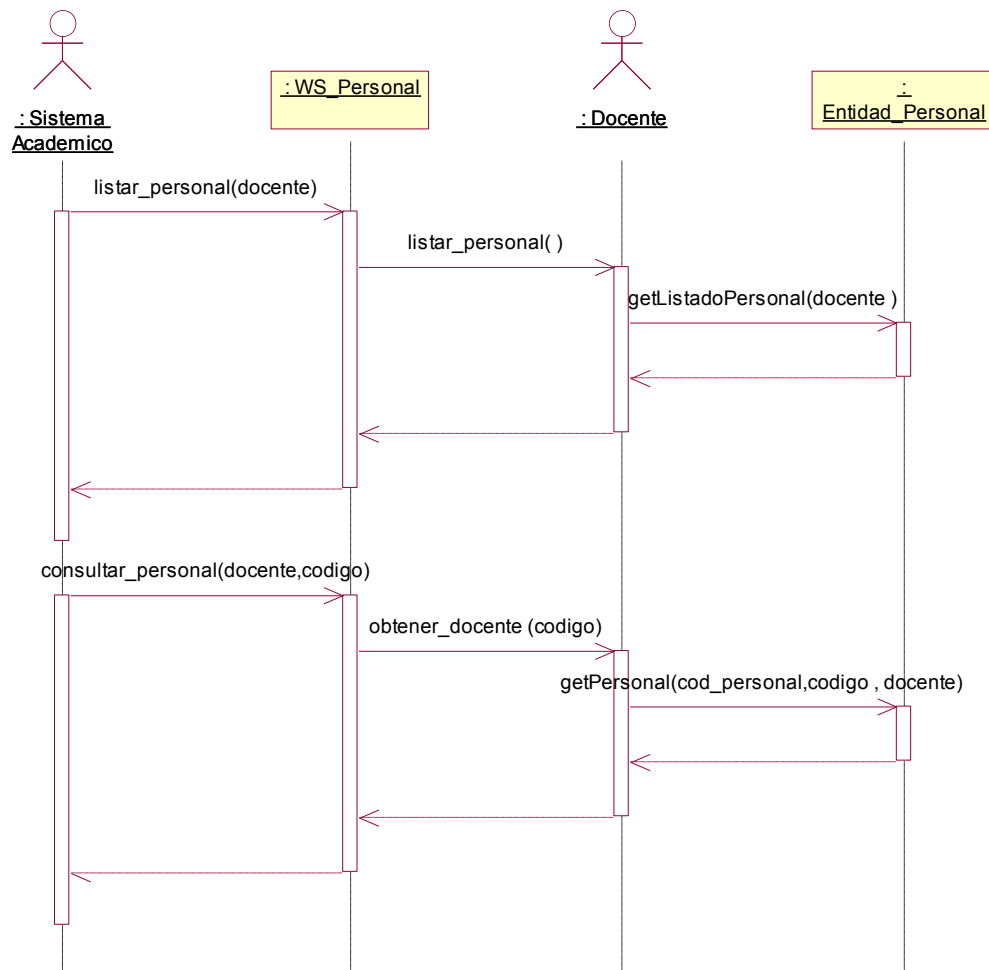


Figura 4.24: Principales diagramas de secuencia del servicio web

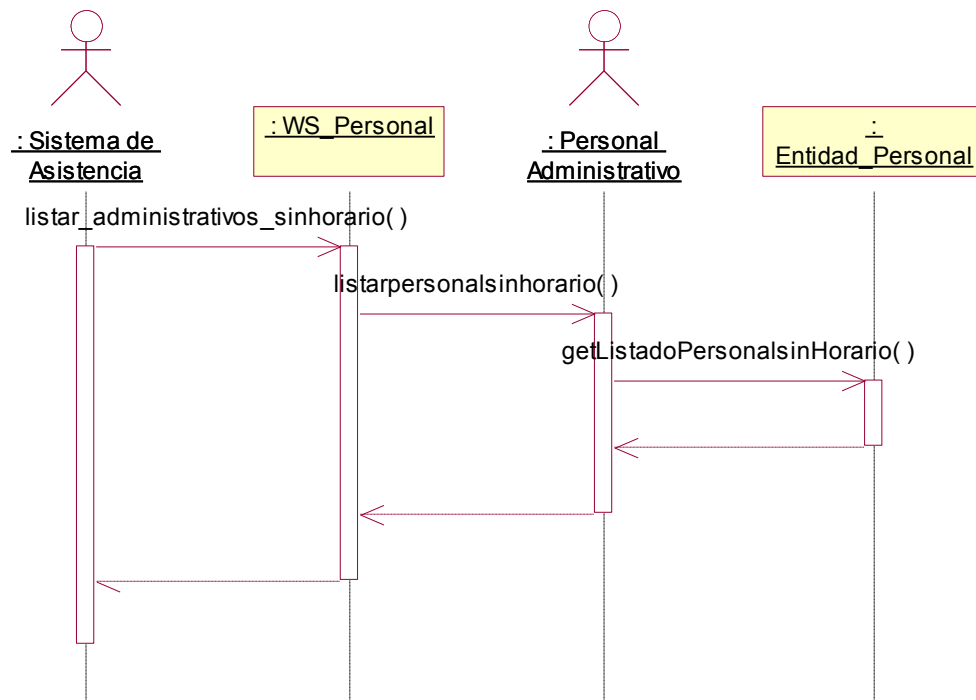


Figura 4.25: Principales diagramas de secuencia del servicio web

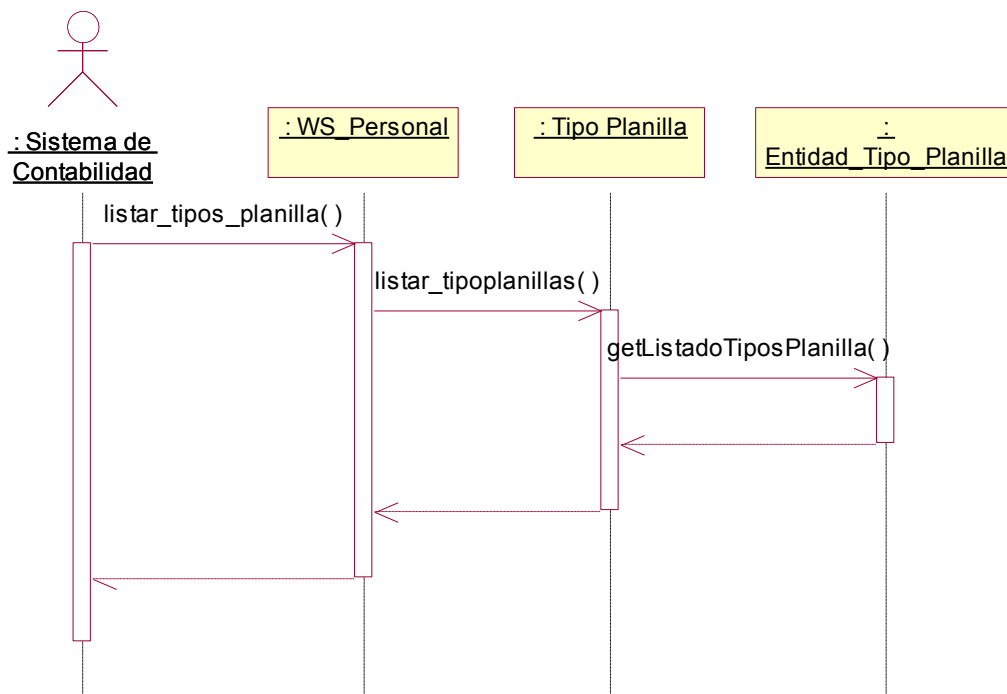


Figura 4.26: Diagrama de secuencia de Listar Tipo Planillas

4.3.6.3 Módulo de Gestión de Alumnos

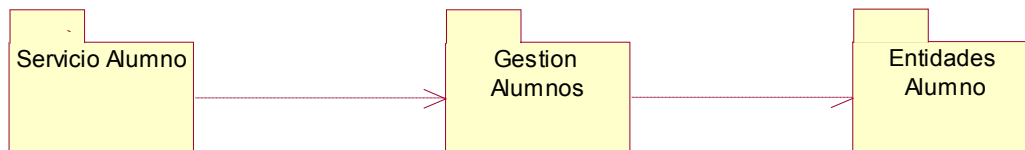


Figura 4.27: Diagrama de paquetes del módulo de gestión de alumnos

Para este módulo se han definido 3 capas, ya que los diferentes sistemas van a acceder al módulo mediante los servicios expuestos. A continuación se detallan las capas:

- Servicio Alumno

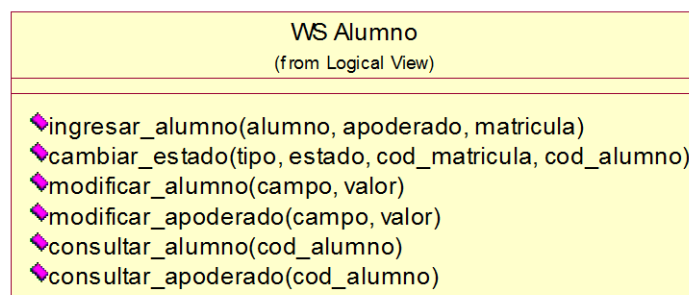


Figura 4.28: Definición del servicio a exponer del módulo de gestión de alumnos

- Gestión Alumnos

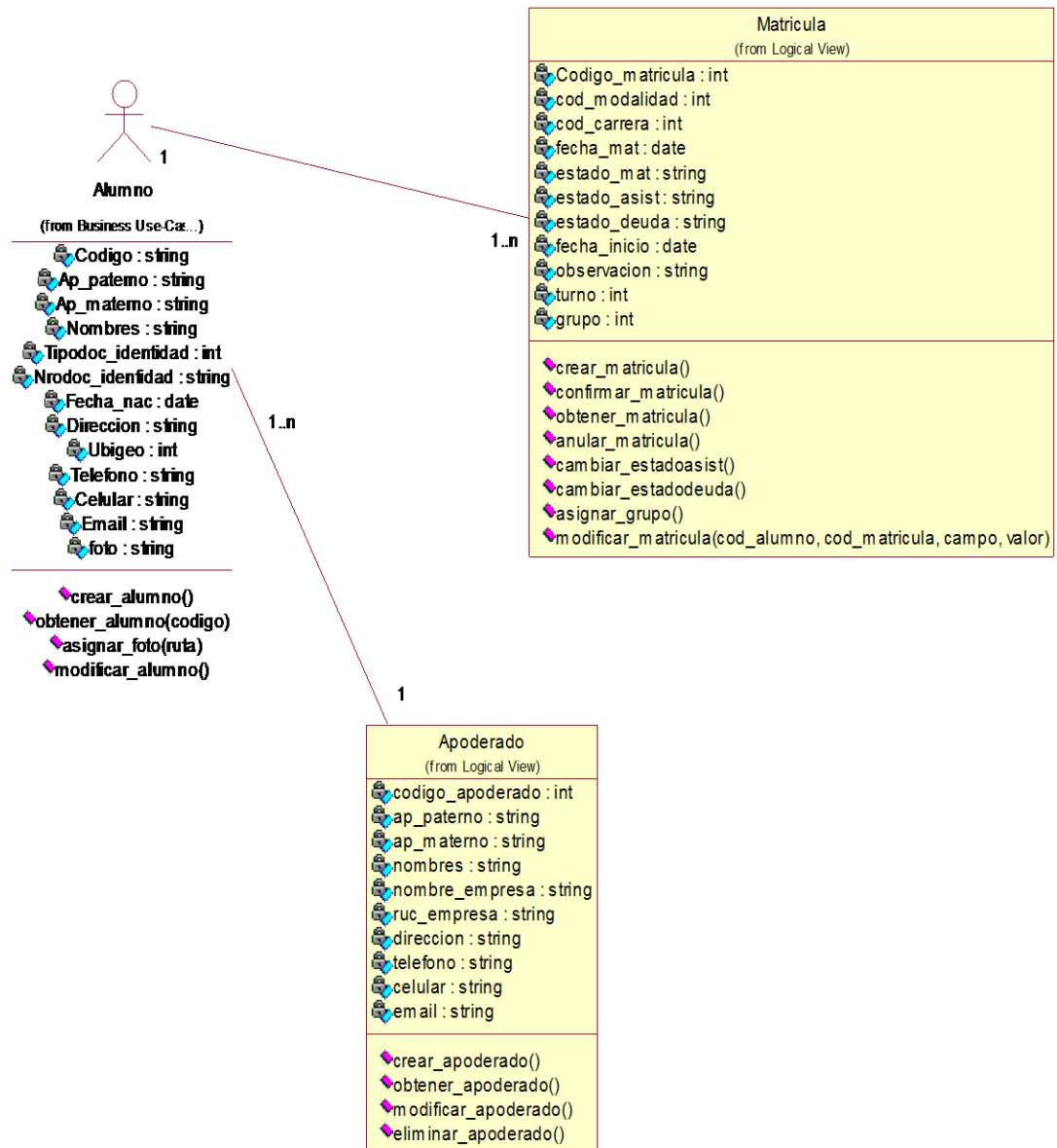


Figura 4.29: Diagrama de clases de la capa de negocios del módulo de gestión de alumnos

- Entidades Alumno

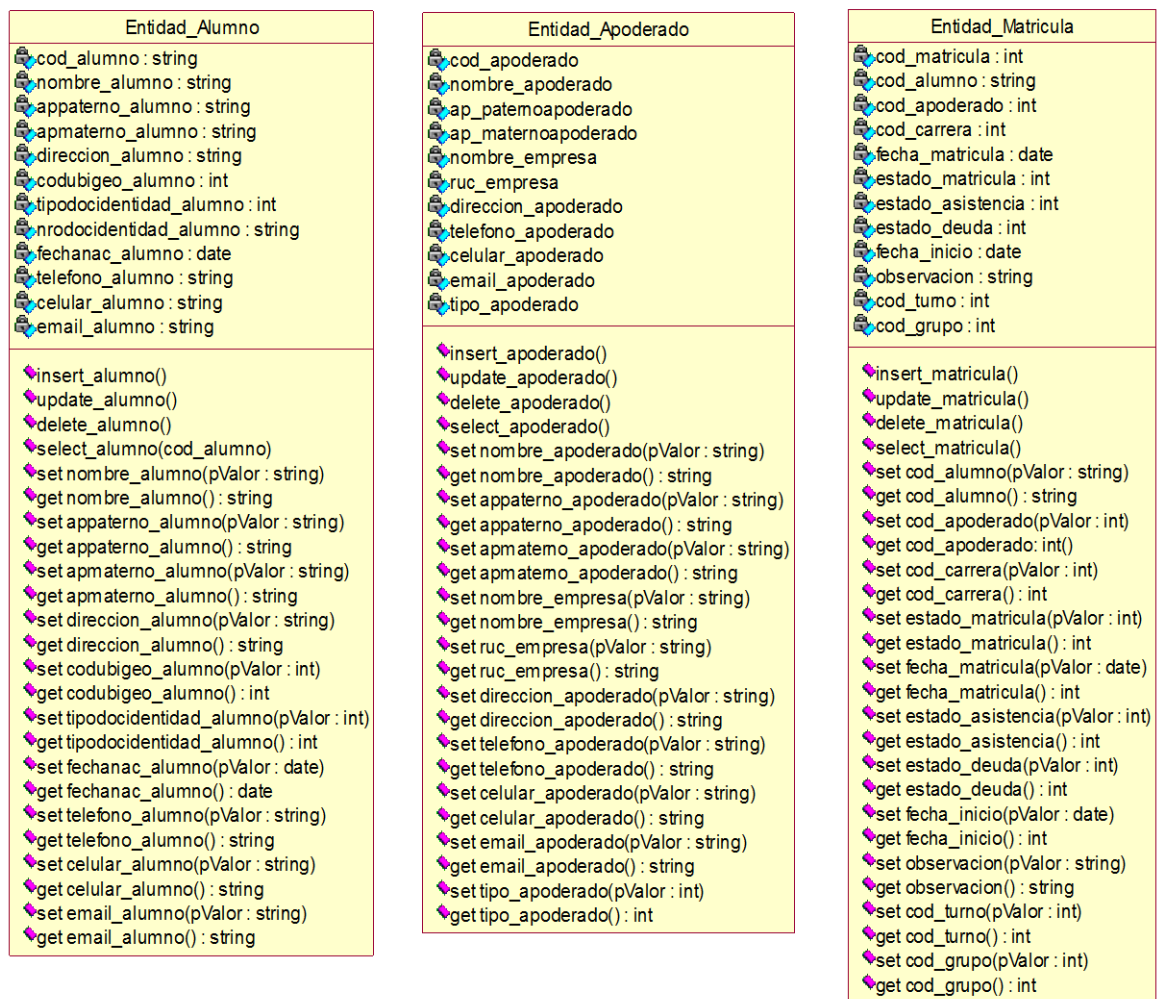


Figura 4.30: Diagrama de Clases entidad del módulo de gestión de alumnos

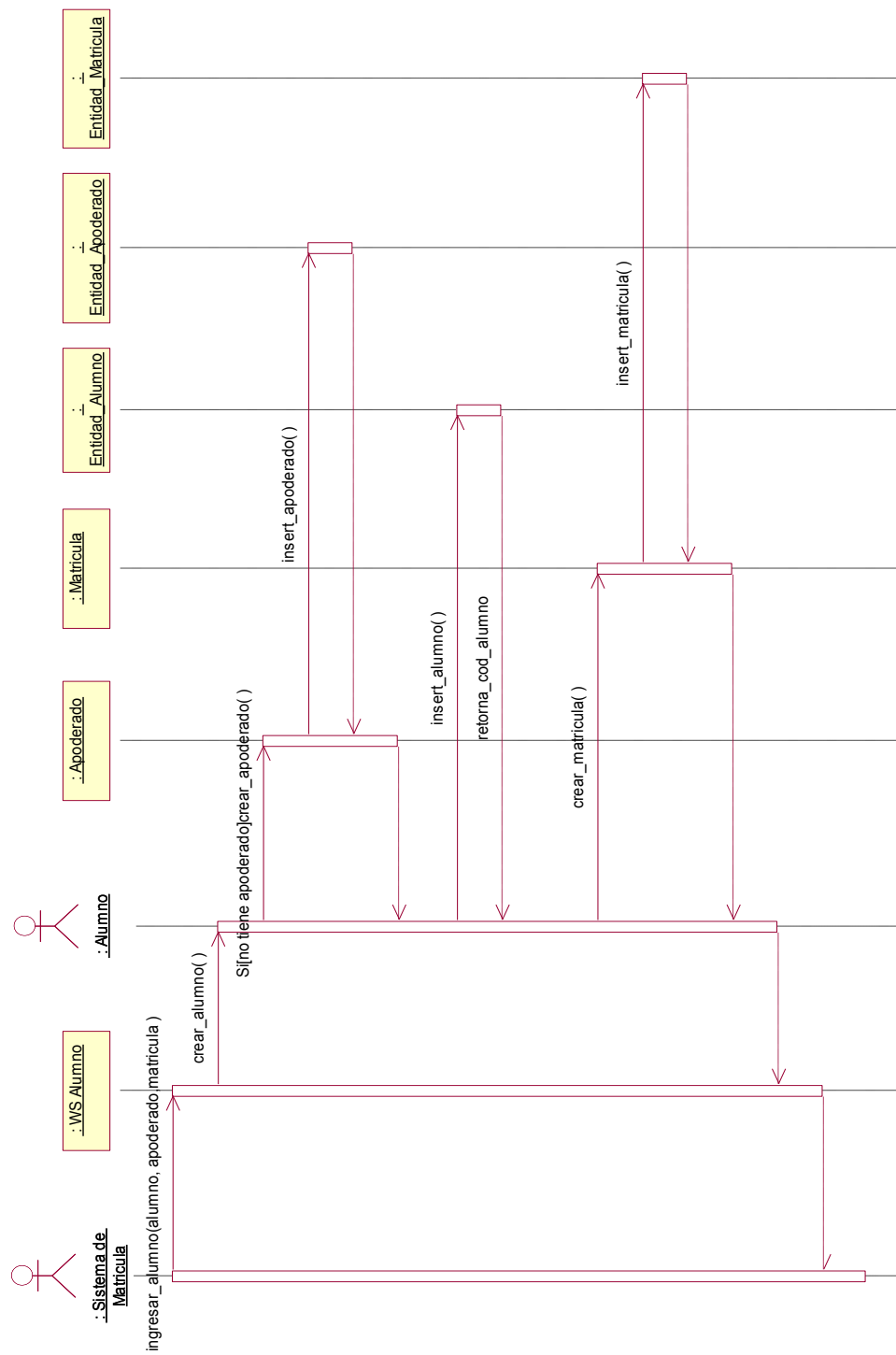


Figura 4.31: Diagrama de secuencia de Crear_Alumno

4.3.6.4 Módulo de consulta de horarios de docentes

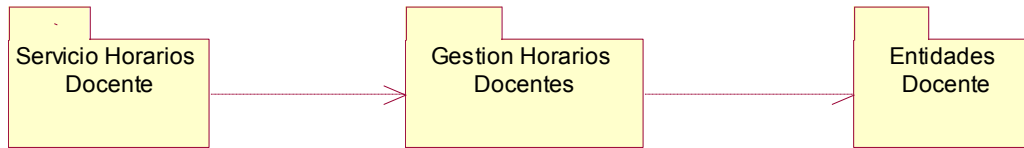


Figura 4.32: Diagrama de paquetes del módulo de consulta de horarios de docentes

Para este módulo se han definido 3 capas, ya que los diferentes sistemas van a acceder al módulo mediante los servicios expuestos. A continuación se detallan las capas:

- Servicio horarios docente

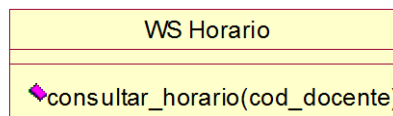


Figura 4.33: Definición del servicio a exponer del módulo de consulta de horarios de docentes

- Gestión horarios docente

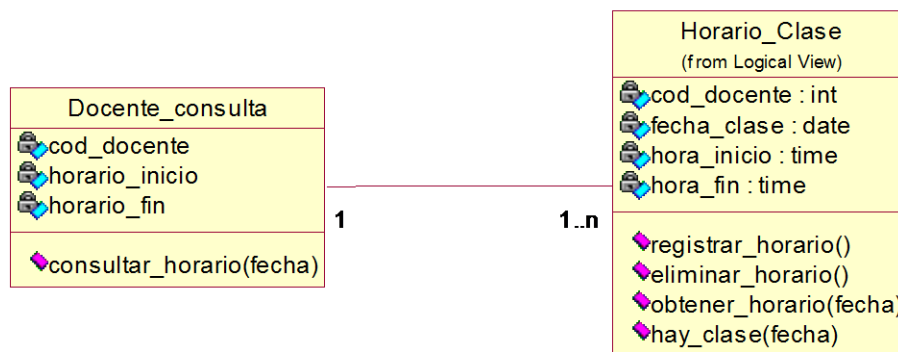


Figura 4.34: Diagrama de clases de la capa de negocios del módulo de consulta de horarios de docentes

- Entidades docente

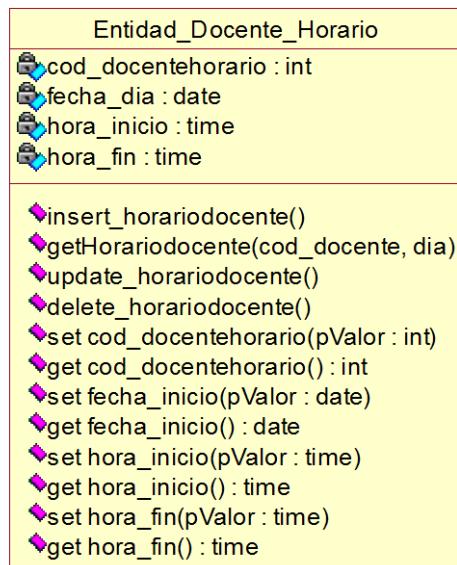


Figura 4.35: Diagrama de Clases de la capa entidad del modulo de consulta de horarios de docente

A continuación se muestra el diagrama de secuencia del caso de uso consultar horario docente:

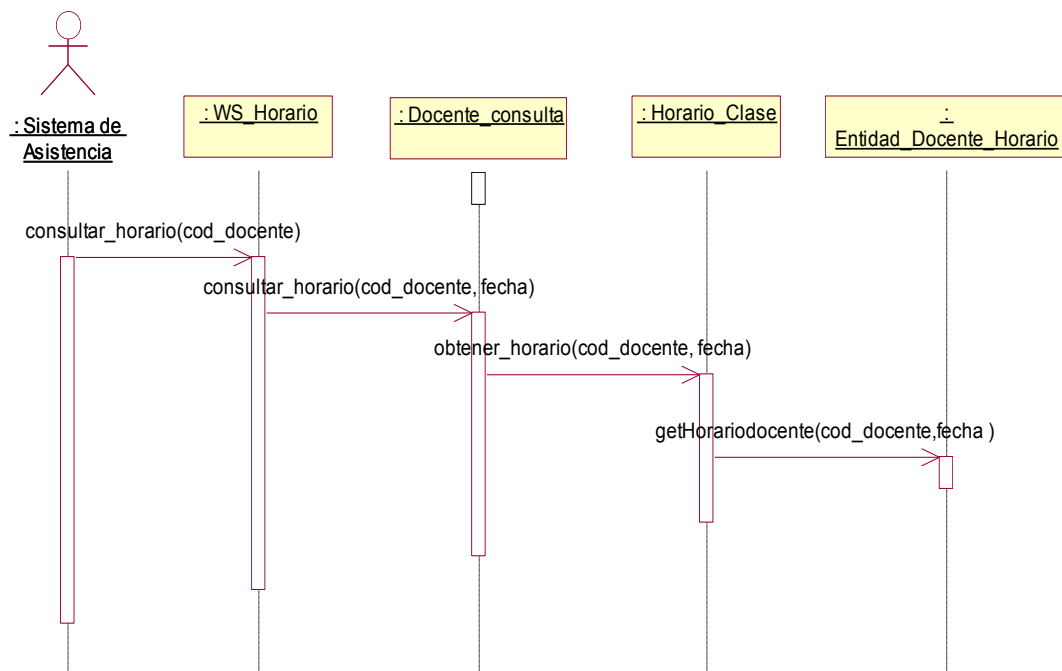


Figura 4.36: Diagrama de secuencia del caso de uso consultar horario docente

4.3.6.5 Módulo de Generación de asientos contables

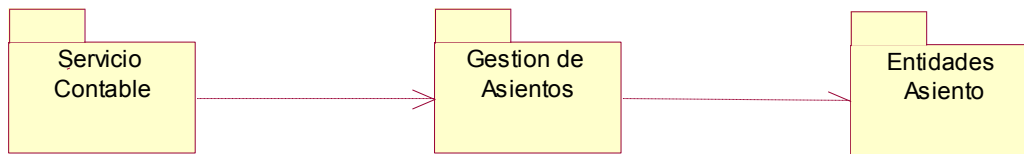
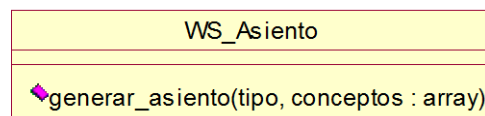


Figura 4.37: Diagrama de paquetes del módulo de generación de asientos contables

Para este módulo se han definido 3 capas, ya que los diferentes sistemas van a acceder al módulo mediante los servicios expuestos. A continuación se detallan las capas:

- Servicio Contable



4.38: Definición del servicio a exponer del módulo de generación de asientos contables

- Gestión de Asientos

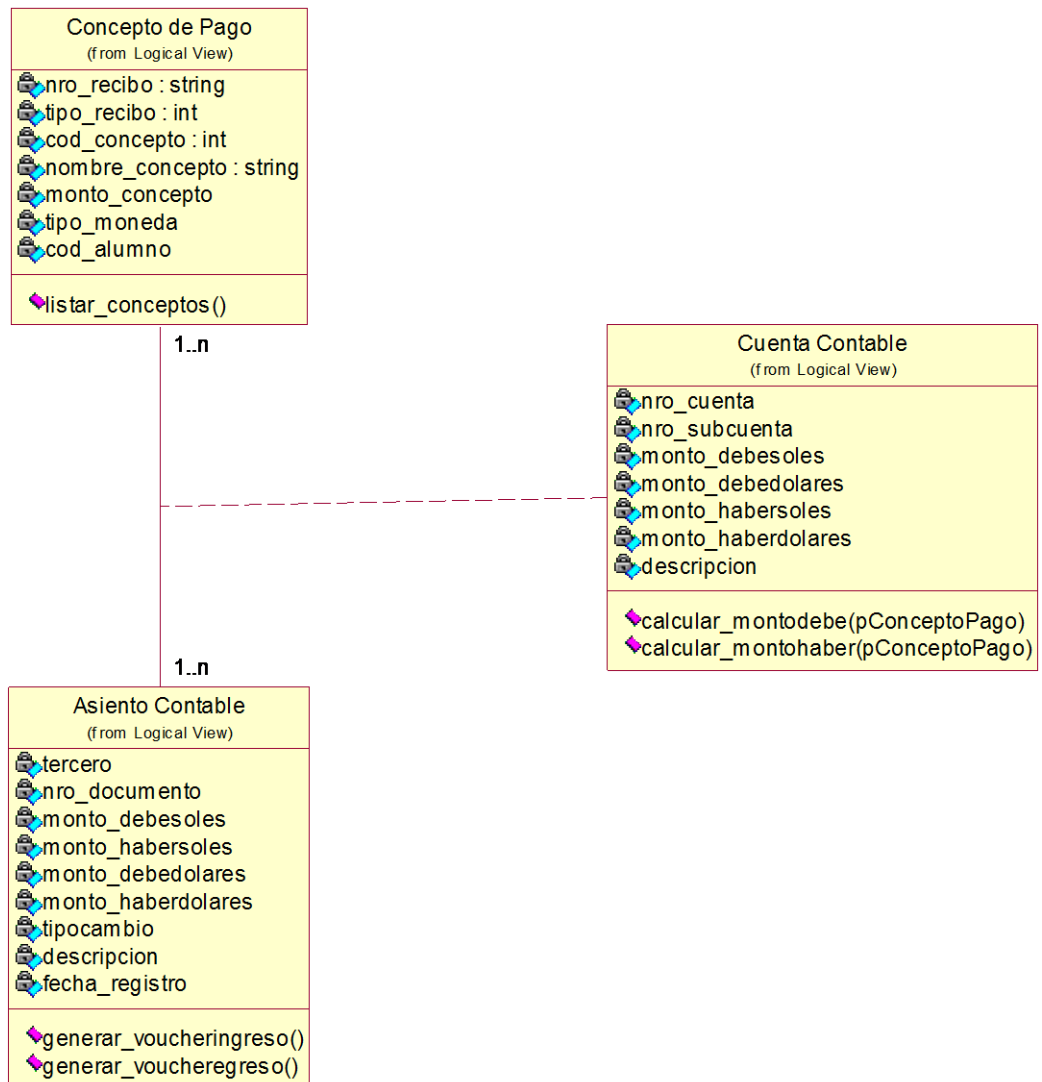


Figura 4.39: Diagrama de clases de la capa de negocios del módulo de generación de asiento contable

- Entidades Asiento

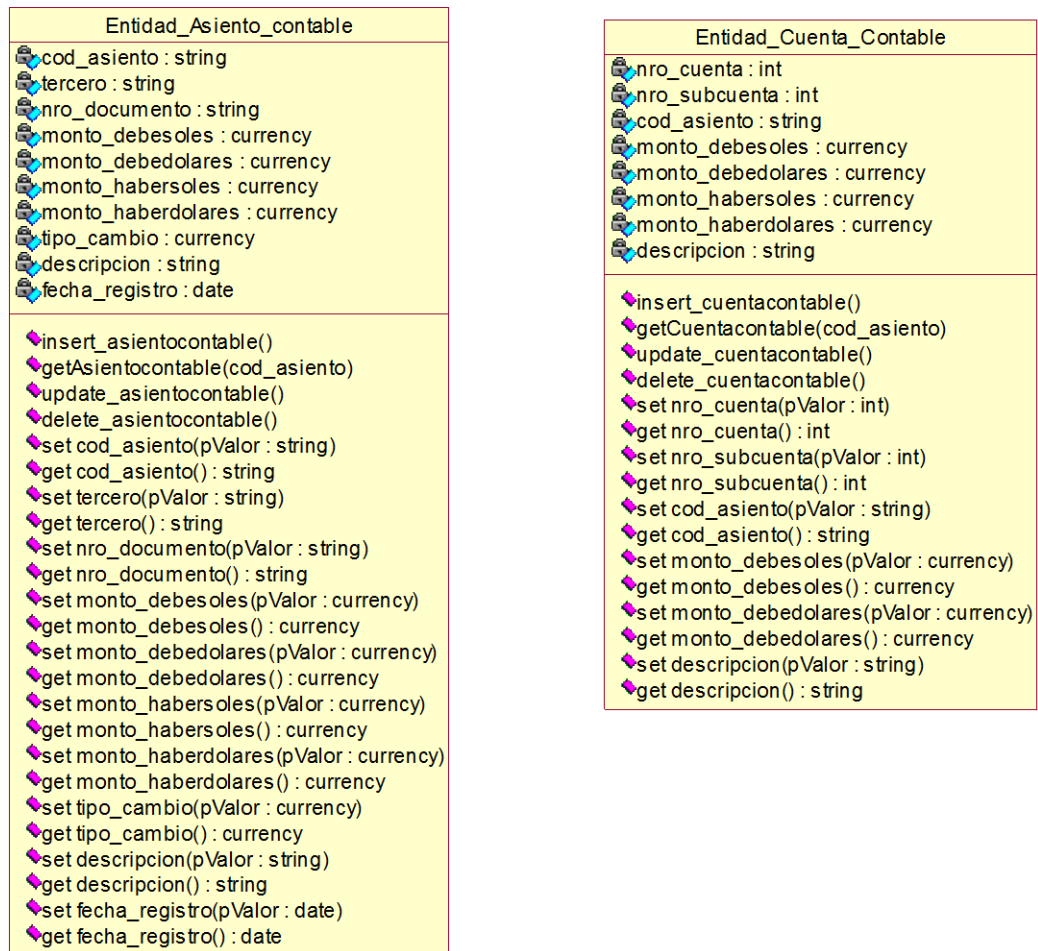


Figura 4.40: Diagrama de clases de la capa entidad del modulo de generación de asiento contable

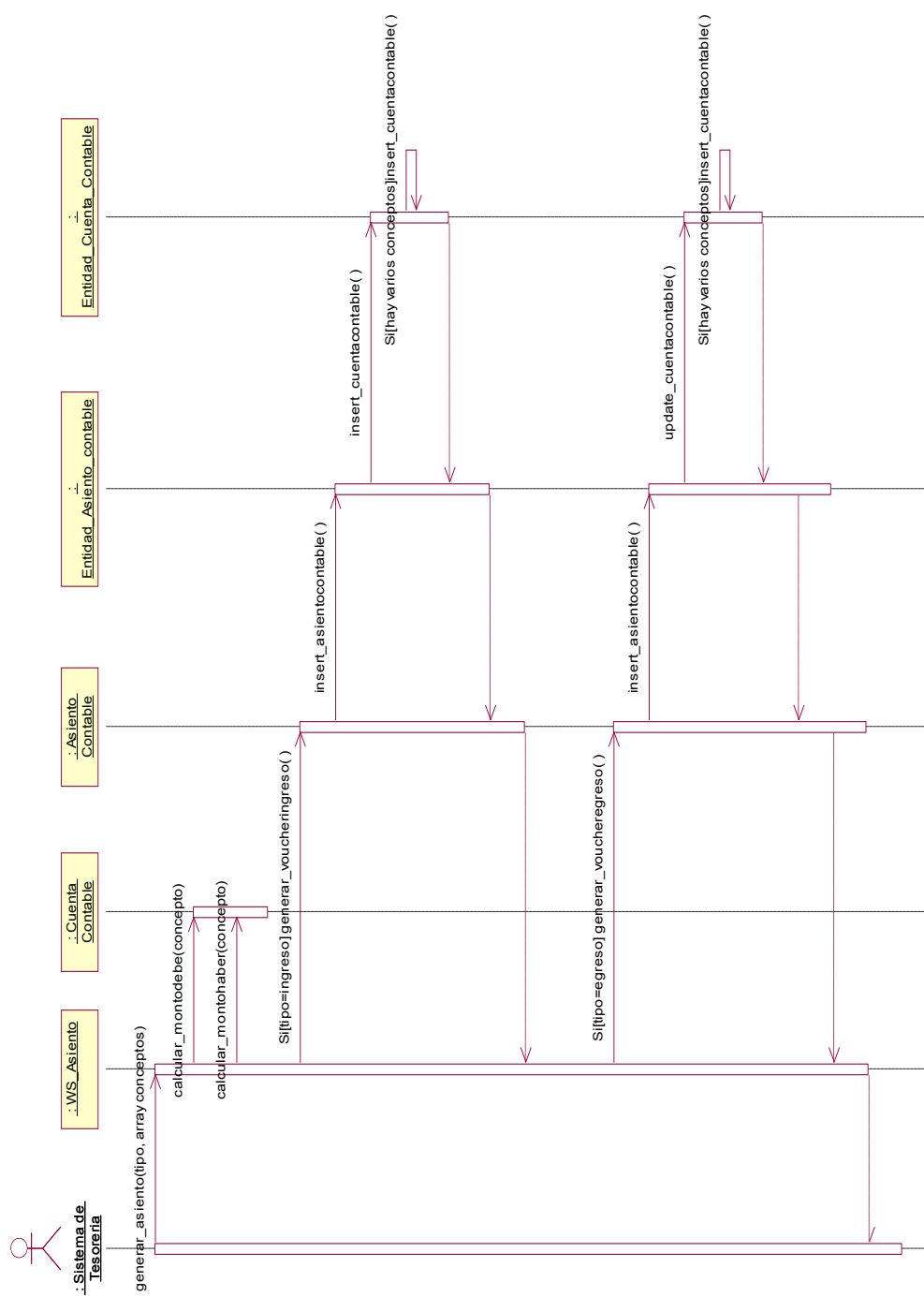


Figura 4.41: Diagrama de secuencia de Generar Asiento Contable

4.3.6.6 Módulo de Gestión de Carreras

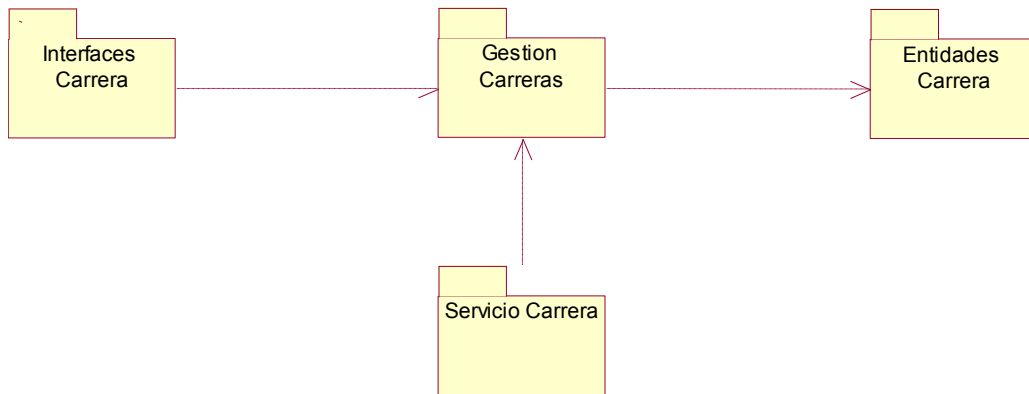


Figura 4.42: Diagrama de paquetes del módulo de gestión de carreras

Se han definido las 4 capas en este módulo, las cuales se detallan a continuación:

- Interfaces Carrera

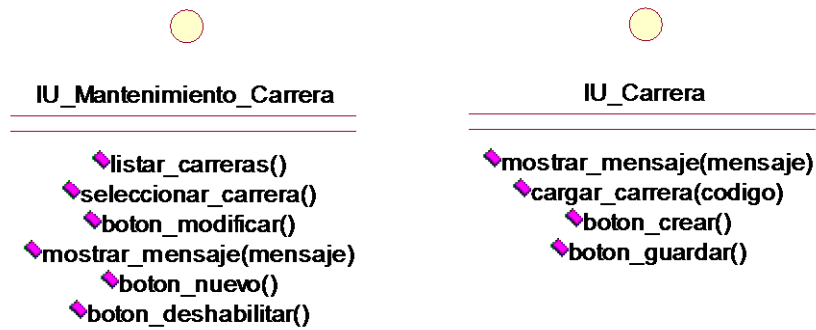


Figura 4.43: Diagrama de Clases interfaz del módulo de gestión de carreras

- Servicio Carrera

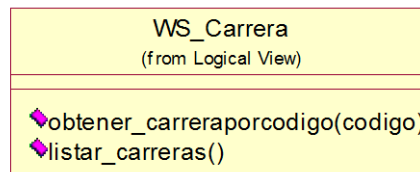


Figura 4.44: Definición del servicio a exponer del módulo de gestión de carreras

- Gestión Carreras

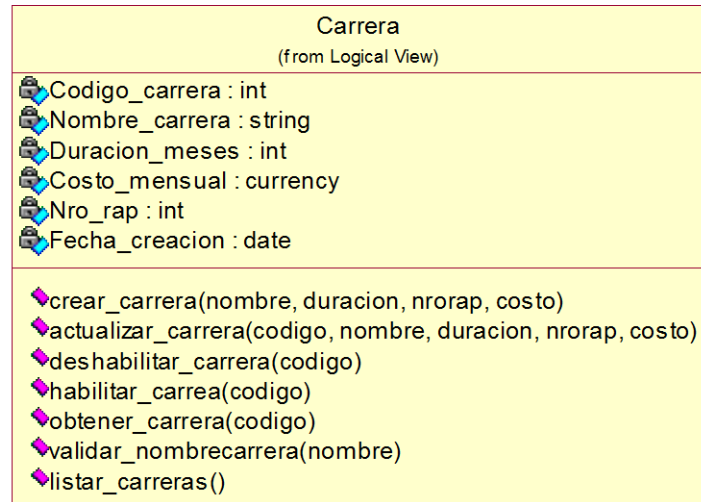


Figura 4.45: Diagrama de clases de la capa de negocios del módulo de gestión de carreras

- Entidades Carrera

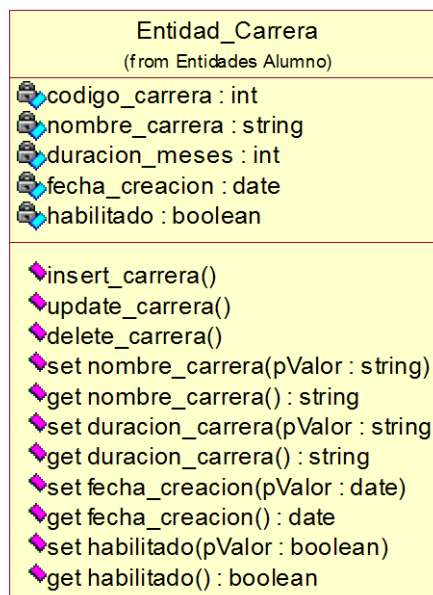


Figura 4.46: Diagrama de Clases entidad del módulo de gestión de carreras

A continuación se muestra el diagrama de secuencia del caso de uso consultar carrera:

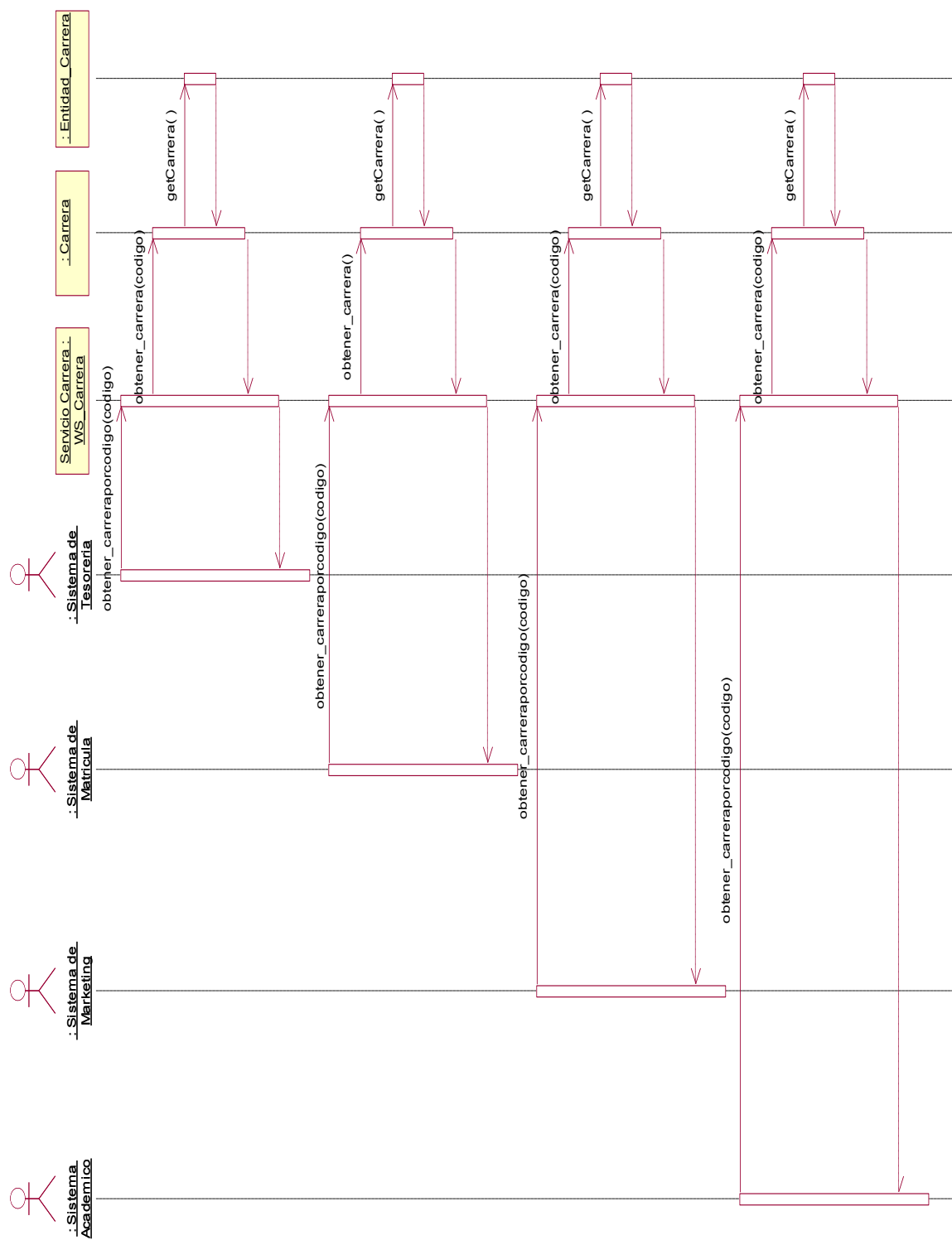


Figura 4.47: Diagrama de secuencia del caso de uso consultar carrera.

4.3.6.7 Módulo de Consolidación de Asistencia de personal y docente

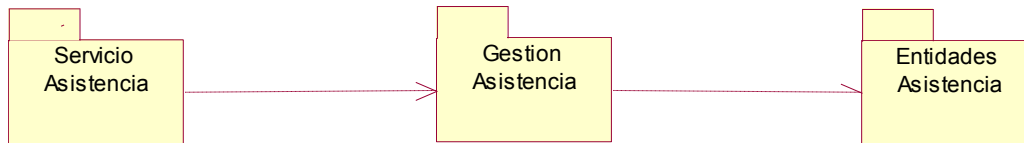


Figura 4.48: Diagrama de paquetes del módulo de consolidación de asistencia de personal y docente

Para este módulo se han definido 3 capas, ya que los diferentes sistemas van a acceder al módulo mediante los servicios expuestos. A continuación se detallan las capas:

- Servicio Asistencia

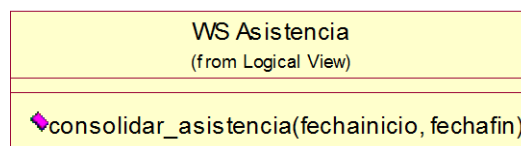


Figura 4.49: Definición del servicio a exponer del módulo de consolidación de asistencia de personal y docente

- Gestión Asistencia

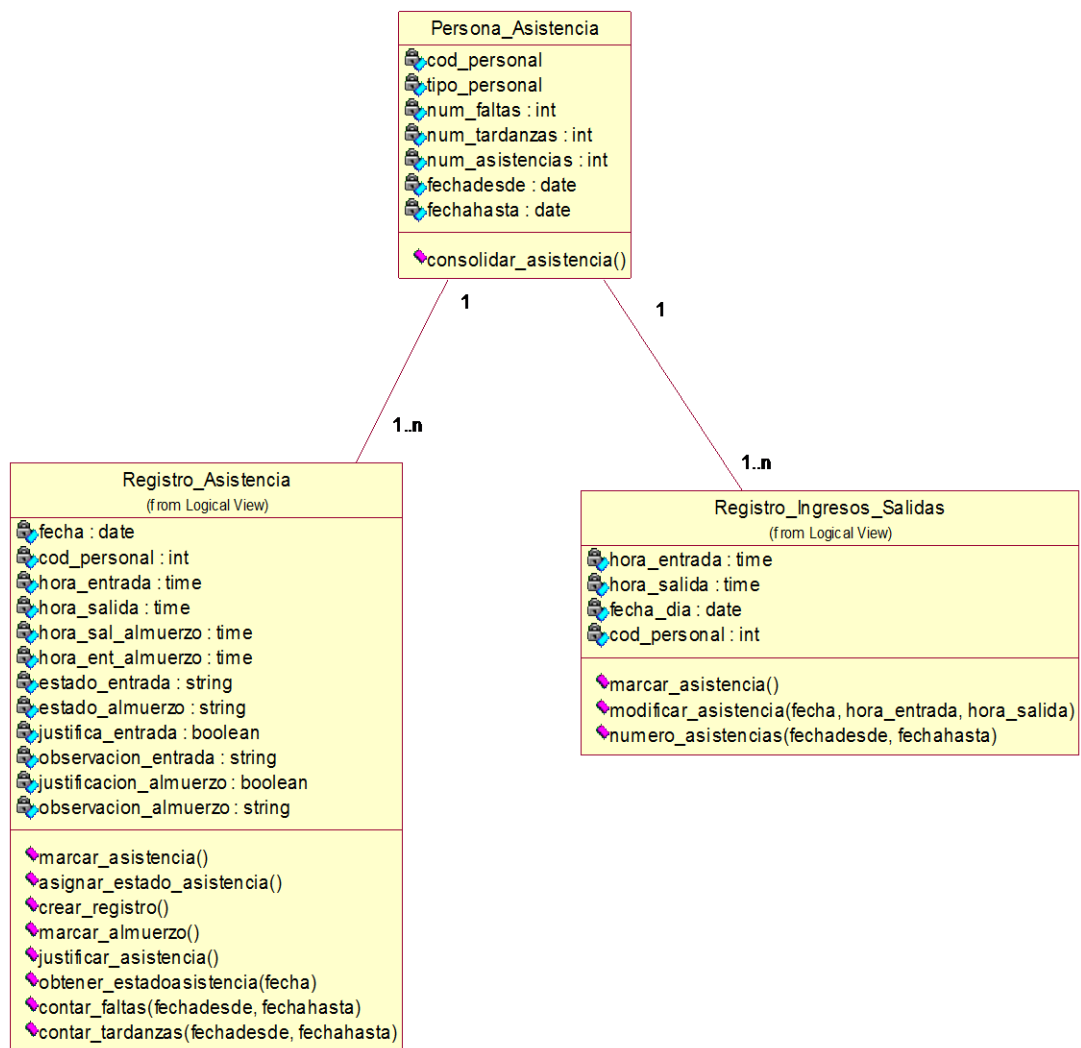


Figura 4.50: Diagrama de clases de la capa de negocios del módulo de consolidación de asistencia de personal y docente

- Entidades Asistencia



Figura 4.51: Diagrama de Clases entidad del módulo de consolidación de asistencia de personal y docentes

A continuación se muestra el diagrama de secuencia del caso de uso consolidar asistencia:

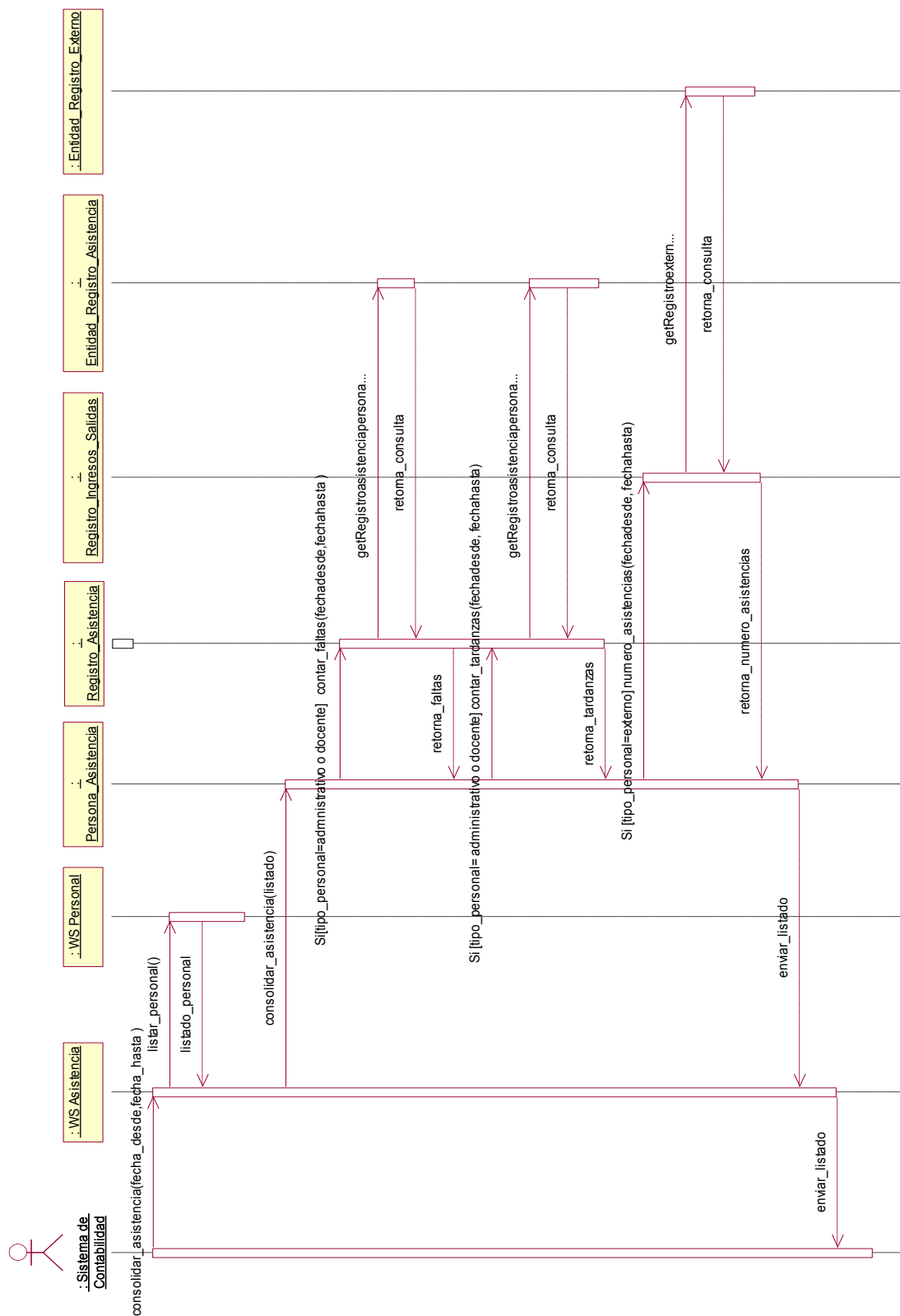


Figura 4.52: Diagrama de secuencia del caso de uso consolidar asistencia

4.3.6.8 Módulo de Gestión de Modalidades

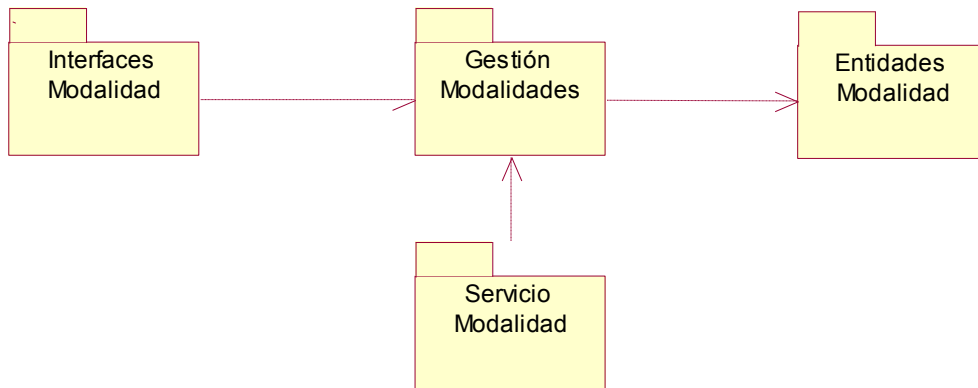


Figura 4.53: Diagrama de paquetes del módulo de gestión de modalidades

Se han definido las 4 capas en este módulo, las cuales se detallan a continuación:

- Interfaces Modalidad



Figura 4.54: Diagrama de Clases interfaz del módulo de gestión de modalidades

- Servicio Modalidad

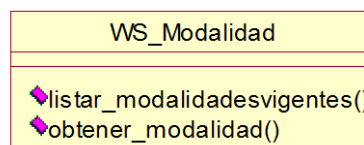


Figura 4.55: Definición del servicio a exponer del módulo de gestión de modalidades

- Gestión Modalidades

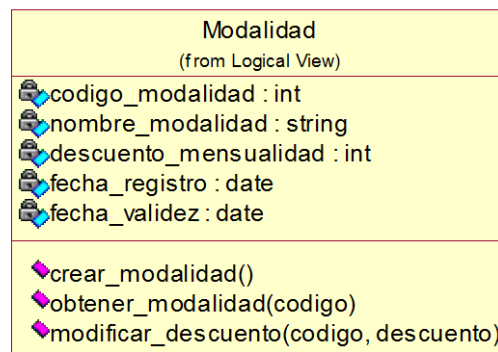


Figura 4.56: Diagrama de clases de la capa de negocio del módulo de gestión de modalidades

- Entidades Modalidad

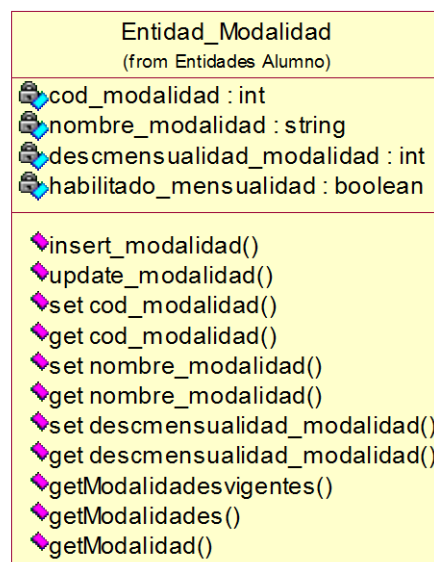


Figura 4.57: Diagrama de Clases entidad del módulo de gestión de modalidades

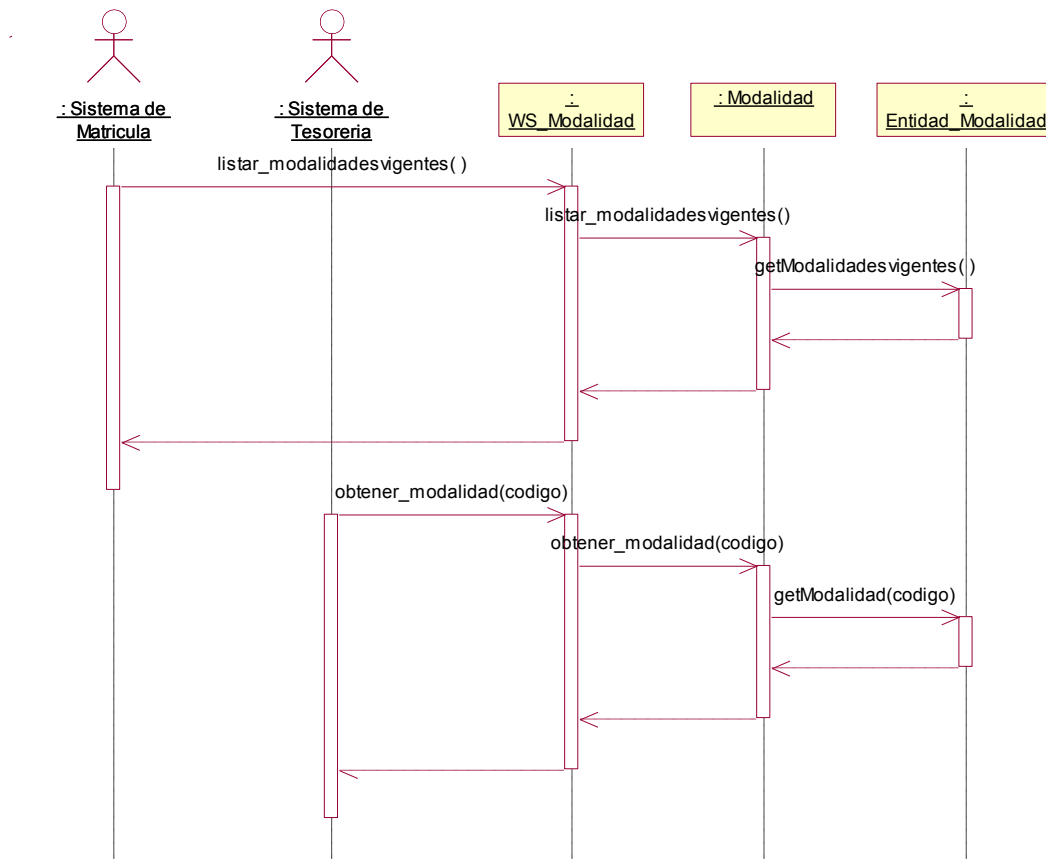


Figura 4.58: Diagrama de secuencia principales del modulo de gestión de modalidades

4.3.7 Vista de Procesos

La solución al problema dado está basada en tecnología Web. Para aquellos módulos que cuentan con la capa de interfaz, ésta corre en la estación de trabajo del usuario en un browser. El browser es el encargado de presentar la interfaz y de enviar al servidor las acciones que el cliente realiza. Por el lado del servidor se encuentra el Internet Information Server (IIS) y el lenguaje a utilizar es ASP.net. El servidor Web recibe las solicitudes del cliente mediante protocolo HTTP, es el que se encarga de gestionar la atención a múltiples clientes en forma simultánea.

Lo relacionado a la distribución de las capas, los servicios expuestos, componentes de negocio y componentes de datos que conforman la solución no se encuentran distribuidos. Las razones de no usar distribución son por performance de las operaciones marcadas como críticas.

Los motores de bases de datos utilizados son 3: SQL Server, Visual Fox Pro y Mysql. Cada uno de ellos corre en forma independiente, atendiendo incluso solicitudes de otras aplicaciones.

Las aplicaciones legadas (Sistema de Matricula, Sistema de Gestión Académica, Sistema de Tesorería, Sistema de Contabilidad y Sistema de Asistencia) fueron adaptados de forma que exponen servicios mediante Web Services.

A continuación se muestra el diagrama de procesos:

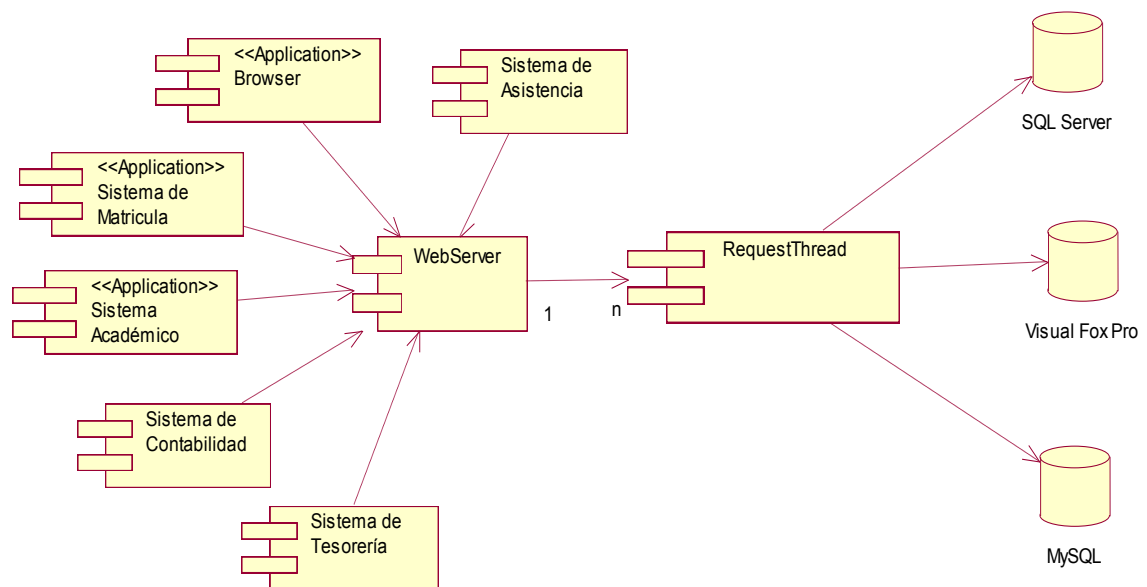


Figura 4.59: Diagrama de la arquitectura de procesos para la solución planteada

Cabe resaltar que la solución planteada no codifica en forma explícita esta arquitectura de procesos, sino que la tecnología a usar brinda servicios de alto nivel que lo encapsulan.

También es importante decir que cuando el servidor Web crea un hilo se asocian a él los contextos transaccionales.

4.3.8 Vista de Implementación

Cada capa de cada servicio fue desarrollada en un componente diferente del resto. Ningún servicio cuenta con interfaces a excepción del Servicio Integrador.

A continuación se muestra el diagrama de componentes que indica la dependencia entre éstos.

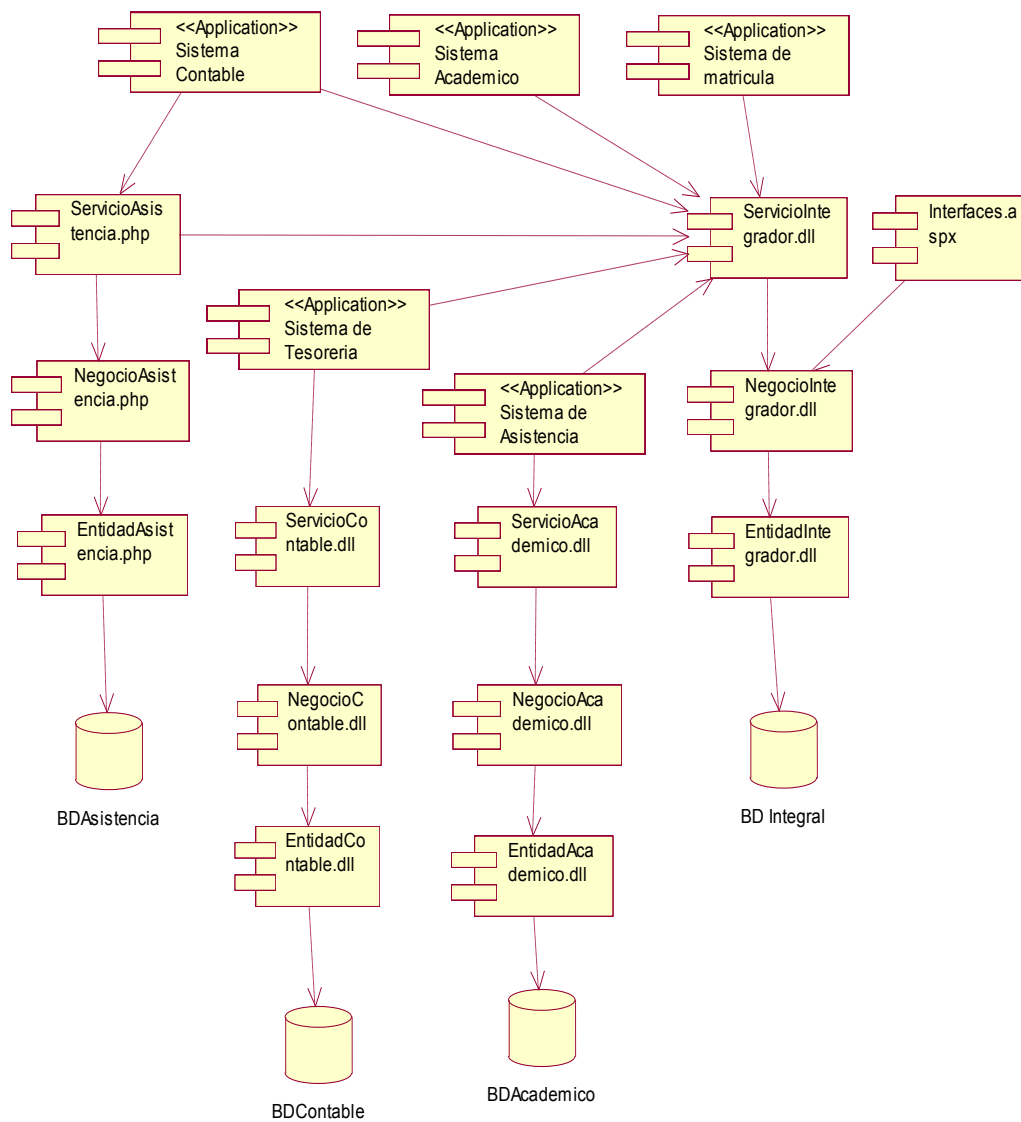


Figura 4.60: Diagrama de componentes para la integración de los sistemas

4.3.9 Vista de Despliegue

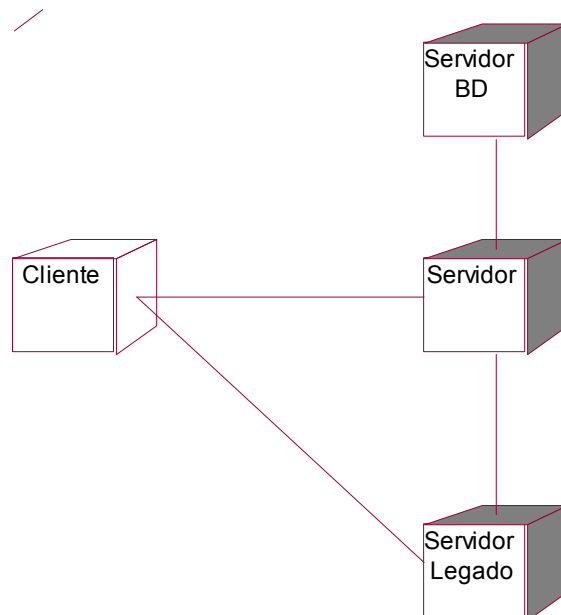


Figura 4.61: Vista de despliegue de la solución de integración

El nodo cliente representa a las estaciones de trabajo de los usuarios finales. El nodo servidor representa al equipo donde correrá el servidor Web y de aplicaciones de los servicios que se exponen. El nodo servidor legado representa la arquitectura informática para correr los sistemas legados en Web y sus servicios expuestos que vendría a ser el Sistema de Asistencia.

Las estaciones de trabajo de los clientes deben de contar con un browser o con la aplicación cliente instalada y tener soporte para el framework de .Net. En el nodo Server debe correr el Internet Information Server y el BizTalk Server donde serán colocados los servicios construidos sobre .Net con sus respectivos assemblies. En el servidor legado se encuentran los servicios Web realizados sobre la plataforma PHP y tiene instalado un servidor Apache.

4.3.10 Atributos de calidad

- Usabilidad: Debido a que se va a exponer funcionalidad de las aplicaciones existentes y modificar éstas para que consuman los servicios que necesitan, ésta transformación va a ser transparente al usuario. No es necesario una capacitación, salvo en los módulos que se están diseñando nuevas interfaces, ya

que los usuario ya están familiarizados con el funcionamiento de las aplicaciones existentes.

- Confiabilidad: Uno de los servicios más críticos de la solución planteada y que no debe fallar es el caso de uso “ingresar alumno”, el cual es utilizado en el momento de la matrícula.
- Performance: El caso de uso “obtener horario docente” debe realizarse en máximo 5 segundos, ya que es utilizado para colocar el estado de la asistencia del docente al momento del marcado en el sistema. El resto de los procesos debe realizarse en un tiempo razonable.

Capítulo 5: Conclusiones y trabajos futuros

5.1 Conclusiones

La integración de sistemas en una organización se ha convertido no sólo en una necesidad operacional sino también estratégica. Las empresas hoy en día necesitan ser cada vez más competitivas para poder surgir en el mercado y para poder lograrlo, uno de los aspectos importantes es que el soporte de la TI se encuentre alineado con las necesidades del negocio.

Hemos visto que la inadecuada planificación de sistemas dentro de una organización puede solucionar los problemas presentes a corto plazo, pero a medida que las organizaciones van creciendo y tienen otras necesidades, una correcta planificación y administración de los recursos de TI es necesaria, de tal manera que la interoperabilidad entre los sistemas existentes en la empresa y los nuevos a implementarse sea posible.

Al utilizar SOA y Web Services para la integración de las aplicaciones legadas en una empresa nos permite reutilizar los recursos existentes y exponerlos en forma de servicios, de tal manera que puedan ser consumidos por las aplicaciones que los necesiten. Estos servicios pueden ser ensamblados u orquestados y las nuevas aplicaciones que se diseñen o se adquieran deberán ser pensadas como servicio.

Con SOA no sólo se busca la integración de aplicaciones a nivel interno de una organización sino que también nos permite la fácil comunicación e integración con los sistemas existentes de otras organizaciones, de este modo una organización puede tener una mejor comunicación con su partner o socio de negocios existente en cualquier lugar del mundo, intercambiando información mediante servicios Web.

5.2 Futuros Trabajos

El enfoque de integración utilizado en la presente tesina es un enfoque de broker de mensajes. La organización en estudio ha dado un primer paso en la adopción de SOA y paulatinamente se van a ir exponiendo más funcionalidades de los sistemas existentes como servicios. De acuerdo a las futuras necesidades puede ser necesario migrar a un enfoque de bus de mensajes.

Actualmente el ingreso de notas y asistencia de los alumnos está centralizada en el departamento académico. Exponiendo estas funcionalidades como una aplicación Web, ya se podría delegar esta responsabilidad a los docentes para que puedan ingresarlos desde donde se encuentren.

Existen ciertas funcionalidades en los sistemas, que al ser expuestos como servicios se van a poder reutilizar para brindar soluciones Web a los clientes de la organización. Entre estas soluciones se podría implementar una intranet para padres, donde ellos puedan consultar las notas de sus hijos, asistencia y el estado de cuenta de sus pagos.

Y por último, en un futuro no muy lejano, una vez que todas las aplicaciones sean adoptadas a un entorno SOA, ya se tendría toda la plataforma necesaria para poder implementar BPM

REFERENCIAS BIBLIOGRÁFICAS

- [Rittammanart07] Rittammanart Nattanicha. Enterprise Application Integration Platform for Corporate Asset Management. Asian Institute of Technology. Master of Science Thesis. (2007) Thailandia.
- [Reldin+07] Reldin Pierre, Sundling Peter. Explaining SOA Service Granularity. Linköping University. Institute of Technology. Master Thesis. (2007) Stockholm.
- [Erl04] Erl Thomas, Service Oriented Architecture: A Field Guide to Integrating XML and Web Services, 2004, ISBN 0-13-1428985, Prentice Hall.
- [High+05] High Rob, Zinder Stephen, Graham Steve, IBM's Soa Foundation: An Architectural Introduction and Overview, 2005
- [Newcomer04] Newcomer Eric, Lomow Greg, Understanding Soa with Web Services, 2004, ISBN 0-321-18086-0, Addison Wesley Professional
- [Bass+03] Bass Len, Clements Paul, Kazman Rick, Software Architecture in Practice, Second Edition, 2003, ISBN 0-321-15495-9, Addison Wesley
- [Gorton06] Gorton Ian, Essential Software Architecture, 2006 ISBN-10 3-540-28713-2, Springer.
- [Kruchten95] Kruchten Philippe, Architectural Blueprints – The “4+1” View Model of Software Architecture, IEEE Software 12, 1995, pp. 42 – 50
- [IEEE00] IEEE Computer Society, IEEE Recommended Practice for Architectural Description of Software-Intensive Systems, IEEE Std 1471-2000
- [Mendoza05] Mendoza Luis, Estudio de Tecnologías Middleware para Sistemas Peer-to-Peer, Universidad Simón Bolívar – Venezuela (2005)
- [Endrei+04] Endrei Mark, Ang Jenny, Arsanjani Ali, Chua Sook, Comte Philippe, Luo Min, Newling Tony, Patterns: Service Oriented Architecture and Web Services, IBM RedBooks, 2004
- [Garlan+94] Garlan David, Shaw Mary. An Introduction to Software Architecture. School of Computer Science Carnegie Mellon University. CMU-CS-94-166, 1994
- [Wahli+07] Wahli Ueli, Ackerman Lee, Di Bari Alessandro, Hodgkinson Gregory, Kerteston Anthony, Olson Laura, Portier Bertrand. Building Soa Solutions using the Rational SDP. IBM Redbooks, 2007.
- [Newcomer+04] Newcomer Eric, Lomow Greg. Understanding SOA with Web Services, 2004, ISBN 0-321-18086-0, Addison Wesley
- [Zhai03] Clay Randy, Application Integration Survey, Integration Consortium, <http://surveys.icmembers.org/document/download/1577>
- [Billy04] Billy Reynoso Carlos, Introducción a la Arquitectura de Software, 2004, http://www.microsoft.com/spanish/msdn/arquitectura/roadmap_arq/intro.mspx
- [Zaragoza06] Universidad de Zaragoza, Conceptos y Estándares de arquitecturas orientadas a servicios web, 2006, Programa de Doctorado

de Informática e Ingeniería de Sistemas,
<http://iaaa.cps.unizar.es/docencia/SWDoct.html>

- [Garcia] García Gonzales Juan Pablo, Pais Wilson, SOA como evolución natural de las soluciones de integración.
<http://desarrollo.datco.cl/MaterialPublico/EvolucionSOA.ppt>
- [Mendoza05] Mendoza Luis, Estudio de Tecnologías Middleware para Sistemas Peer-to-Peer, 2005, Universidad Simón Bolívar, Laboratorio de Investigación en Sistemas de Información.
http://www.lisi.usb.ve/publicaciones/07%20integracion%20de%20sistemas/integracion_17.pdf
- [Sessions07] Sessions Roger, A Comparison of the Top Four Enterprise-Architecture Methodologies, MSDN Architecture Center (2007)
<http://msdn.microsoft.com/en-us/library/bb466232.aspx>
- [Bray97] Bray Mike, Middleware, Carnegie Mellon Institute, (1997),
<http://www.sei.cmu.edu/str/descriptions/middleware.html>
- [Alvarez07] Alvarez Pedro, Bañares Jose Angel, Conceptos y estándares de arquitecturas orientadas a servicios web, Universidad de Zaragoza (2007),
<http://iaaa.cps.unizar.es/docencia/SWD/2.Middleware.pdf>
- [Colan04] Colan Mark, Service-Oriented Architecture expands the vision of Web Services, Part 1, IBM Corporation, 2004,
<http://www.ibm.com/developerworks/webservices/library/ws-soaintro.html>
- [Microsoft04] Integration Topologies, Patterns & Practices, 2004
<http://msdn.microsoft.com/en-us/library/ms978718.aspx>
- [Holley+03] Holley Kerrie, Channabasavaiah Kishore, Tuggle Edward, Migrating to a Service-oriented architecture, Part 1, 2003
<http://www.ibm.com/developerworks/webservices/library/ws-migratesoa/>
- [Tuggle+03] Tuggle Edward, Holley Kerrie, Channabasavaiah Kishore, Migrating to a Service-oriented architecture, Part 2, 2003
<http://www.ibm.com/developerworks/webservices/library/ws-migratesoa2/>
- [Zachman08] Zachman John, Zachman Framework for Enterprise Architecture.
http://www.zachmaninternational.com/2/Zachman_Framework.asp
- [Reynoso] Reynoso Billy, Arquitectura Orientada a Servicios, Universidad de Buenos Aires. <http://download.microsoft.com/download/4/F/F/4FF88340-43CC-4C5B-8E50-09002969D0DD/20051129-ARC-BA.ppt>
- [Karastoyanova+03] Karastoyanova Dimka, Buchmann Alejandro. Components, Middleware and Web Services. Portugal 2003. <http://www.dvs.tu-darmstadt.de/publications/pdf/IADIS-ICWI-2003.pdf>

Anexos

Especificaciones de los Casos de Uso

Nombre:	Crear usuario
Actores:	Administrador del Sistema
Descripción:	El actor crea una cuenta para un nuevo usuario.
Precondiciones:	Ninguna
Flujo Normal:	1. El actor selecciona el botón Crear Usuario. 2. El actor ingresa el nombre, apellido paterno y apellido materno del nuevo usuario. 3. El actor ingresa el nombre de usuario y password correspondiente al nuevo usuario. 4. Selecciona el botón Crear.
Flujo Alternativo	Ninguno.
PostCondiciones	El actor debe asignarle los permisos y accesos a los sistemas correspondientes.

Nombre:	Modificar usuario
Actores:	Administrador del Sistema
Descripción:	El actor modifica los datos de la cuenta de un usuario seleccionado.
Precondiciones:	

El usuario debe haber sido creado previamente.
Flujo Normal: 1. El actor selecciona un usuario de la lista. 2. El actor selecciona el botón Modificar Usuario. 3. El actor modifica los datos del usuario seleccionado. 4. Selecciona el botón Grabar.
Flujo Alternativo 3'. No se quiere modificar los datos del usuario sino los accesos. 4'. Selecciona el botón Modificar Accesos.
PostCondiciones El actor puede modificar los permisos y accesos a los sistemas correspondientes.

Nombre:	Eliminar usuario
Actores:	Administrador del Sistema
Descripción:	El actor elimina un usuario creado anteriormente.
Precondiciones:	El usuario debe haber sido creado previamente.
Flujo Normal:	1. El administrador selecciona un usuario de la lista. 2. El administrador selecciona el botón Eliminar Usuario. 3. El administrador acepta la confirmación de la cancelación.
Flujo Alternativo	3'. El administrador no acepta la confirmación de la cancelación y no se procede con la eliminación.
PostCondiciones	Ninguna.

Nombre:	Gestionar accesos
Actores:	Administrador del Sistema
Descripción:	El actor desea gestionar los accesos a los sistemas de un usuario creado.
Precondiciones:	El usuario debe haber sido creado previamente o ya existir registrado.
Flujo Normal:	1. Se muestran los sistemas y los tipos de accesos. 2. Se muestran los accesos asignados actualmente al usuario seleccionado. 3. Se presentan los botones de crear, modificar y eliminar acceso.
Flujo Alternativo	2'. No muestra ningún acceso debido a que se ha creado el usuario recientemente.
PostCondiciones	Ninguna.

Nombre:	Crear Acceso
Actores:	Administrador del Sistema
Descripción:	El actor asigna el tipo de acceso a un sistema del usuario seleccionado.
Precondiciones:	El usuario debe haber sido creado previamente o ya existir registrado. Se debe haber ingresado a la opción Modificar Usuario y Gestionar Accesos.
Flujo Normal:	1. Se selecciona el sistema al cual se desea dar acceso al usuario. 2. Se selecciona los menús del sistema a los cuales se desea dar acceso y por cada uno de ellos se le asigna el tipo de acceso. 3. Se selecciona el botón Guardar.
Flujo Alternativo	

Ninguno.
PostCondiciones Ninguna.

Nombre:	Modificar acceso
Actores:	Administrador del Sistema
Descripción:	El actor modifica un tipo de acceso a un menú de un sistema para el usuario seleccionado.
Precondiciones:	Debe haberse creado un acceso al menú que se desea modificar.
Flujo Normal:	1. Se selecciona el sistema de la lista. 2. Aparecen los menús de ese sistema a los cuales el usuario tiene acceso. 3. Selecciona el menú al cual se quiere modificar el tipo de acceso. 4. Selecciona de la lista otro tipo de acceso para el menú seleccionado. 5. Selecciona el botón Guardar.
Flujo Alternativo	Ninguno.
PostCondiciones	Ninguna.

Nombre:	Eliminar acceso
Actores:	Administrador del Sistema
Descripción:	El actor elimina el acceso a un menú de un sistema para el usuario seleccionado.
Precondiciones:	Debe haberse creado un acceso al menú que se desea eliminar.

Flujo Normal: 1. Se selecciona el sistema de la lista. 2. Aparecen los menús de ese sistema a los cuales el usuario tiene acceso. 3. Selecciona el menú al cual se quiere eliminar el acceso. 4. Selecciona el botón Eliminar.
Flujo Alternativo Ninguno.
PostCondiciones Ninguna.

Nombre:	Validar Usuario
Actores:	Sistema Académico, Sistema de Contabilidad, Sistema de Asistencia, Sistema de Tesorería, Sistema de Matrícula.
Descripción:	El actor solicita la validación de usuario.
Precondiciones:	Ninguna.
Flujo Normal:	1. El actor envía los datos de ingreso del usuario. 2. El servicio hace la validación correspondiente y retorna los accesos respectivos al actor que lo solicitó.
Flujo Alternativo	2'. En caso de no existir el acceso del usuario al sistema correspondiente, retorna un mensaje de error.
PostCondiciones	Ninguna.

Nombre:	Crear nuevo personal
Actores:	Administrador del Sistema
Descripción:	El actor crea un nuevo personal.
Precondiciones:	No debe estar registrado previamente.
Flujo Normal:	1. Se ingresan todos los datos personales del nuevo personal que se está registrando. 2. Se selecciona el área al cual pertenece. 3. Se selecciona el tipo de personal que es. 3. Se selecciona el botón guardar.
Flujo Alternativo	1'. Si no se ingresan todos los datos del nuevo personal se muestra un mensaje de error.
PostCondiciones	Queda como pendiente la creación de sus horarios en el sistema correspondiente dependiendo del tipo de personal que sea.

Nombre:	Modificar personal
Actores:	Administrador del Sistema
Descripción:	El actor modifica los datos de personal seleccionado.
Precondiciones:	El personal debe estar registrado previamente.
Flujo Normal:	1. Se modifican los datos personales del personal seleccionado. 2. Se selecciona el botón Guardar.
Flujo Alternativo	

1'. Se modifica el área al cual pertenece el personal seleccionado.
1'. Se modifica el tipo de personal asignado al personal seleccionado.
PostCondiciones Ninguna.

Nombre:	Eliminar personal
Actores:	Administrador del Sistema
Descripción:	El actor elimina a un personal seleccionado.
Precondiciones:	El personal debe estar registrado previamente.
Flujo Normal:	1. Selecciona el personal que desea eliminar de la lista. 2. Se selecciona el botón Eliminar. 3. Acepta la confirmación de la eliminación.
Flujo Alternativo	3'. Si no acepta la confirmación de la eliminación no se procede.
PostCondiciones	Ninguna.

Nombre:	Asignar tipo planilla
Actores:	Administrador del Sistema
Descripción:	El actor asigna el tipo de planilla correspondiente al personal seleccionado.
Precondiciones:	Ninguna.
Flujo Normal:	1. Si el personal recién ha sido creado, se selecciona el botón asignar planilla

de la ventana nuevo personal.
2. Se selecciona de la lista el tipo de planilla correspondiente al personal seleccionado.
3. Selecciona el botón Aceptar.
Flujo Alternativo
1'. Si el personal ya tiene asignado un tipo de planilla, se selecciona el usuario de la lista y se ingresa a la opción Modificar personal.
3'. Si se selecciona el botón Cancelar, se anula el registro del tipo de planilla.
PostCondiciones
Ninguna.

Nombre:	Crear tipo de planilla
Actores:	Administrador del Sistema
Descripción:	El actor crea un tipo de planilla para el personal.
Precondiciones:	Ninguna.
Flujo Normal:	1. Se ingresan los datos correspondientes al tipo de planilla. 2. Se selecciona el botón Crear.
Flujo Alternativo	2'. Si la validación de los datos es incorrecta se muestra un mensaje de error.
PostCondiciones	Ninguna.

Nombre:	Modificar tipo de planilla
Actores:	Administrador del Sistema
Descripción:	El actor desea modificar los datos de algún tipo de planilla creada previamente.
Precondiciones:	El tipo de planilla debe estar creado previamente.
Flujo Normal:	1. Se modifican los datos del tipo de planilla seleccionado. 2. Se selecciona el botón Guardar.
Flujo Alternativo	2'. Si los datos ingresados no cumplen con la validación se muestra un mensaje de error.
PostCondiciones	Ninguna.

Nombre:	Crear área
Actores:	Administrador del Sistema
Descripción:	El administrador desea crear una nueva área.
Precondiciones:	Ninguna.
Flujo Normal:	1. Se ingresan los datos correspondientes a la nueva área. 2. Se selecciona el botón Crear.

Flujo Alternativo	
2'. Si los datos ingresados no cumplen con la validación se muestra un mensaje de error.	
PostCondiciones	
Ninguna.	

Nombre:	Modificar área
Actores:	
Administrador del Sistema	
Descripción:	
El actor desea crear una nueva área.	
Precondiciones:	
Ninguna.	
Flujo Normal:	
1. Se ingresan los datos correspondientes a la nueva área.	
2. Se selecciona el botón Crear.	
Flujo Alternativo	
2'. Si los datos ingresados no cumplen con la validación se muestra un mensaje de error.	
PostCondiciones	
Ninguna.	

Nombre:	Eliminar área
Actores:	
Administrador del Sistema	
Descripción:	
El actor desea eliminar un área creada.	
Precondiciones:	
No debe de haber algún personal que pertenezca a esa área.	

Flujo Normal: 1. Selecciona el área de la lista correspondiente. 2. Se selecciona el botón Eliminar. 3. Se confirma la eliminación.
Flujo Alternativo 3'. Se cancela la eliminación.
PostCondiciones Ninguna.

Nombre:	Consultar datos del personal
Actores:	Sistema Académico, Sistema de Contabilidad, Sistema de Tesorería, Sistema de Asistencia.
Descripción:	El actor desea consultar los datos del personal seleccionado.
Precondiciones:	Ninguna.
Flujo Normal:	1. El actor envía los datos de identificación del personal del cual necesita consultar sus datos. 2. Se le retorna al actor los datos solicitados.
Flujo Alternativo	2'. En caso de que los datos de identificación no sean válidos se le envía un mensaje de error.
PostCondiciones	Ninguna.

Nombre:	Consultar tipo planillas
Actores:	Sistema de Contabilidad
Descripción:	El actor desea consultar los tipos de planillas creados.
Precondiciones:	Ninguna.
Flujo Normal:	1. El actor solicita el envío de los datos de los tipos de planillas que existen. 2. Se le retorna al actor los datos solicitados.
Flujo Alternativo	2'. En caso de que no exista ningún tipo de planilla creado se le enviará un mensaje de notificación.
PostCondiciones	Ninguna.

Nombre:	Ingresar alumno
Actores:	Sistema de Matrícula
Descripción:	El actor envía los datos del alumno, el apoderado y la matricula para ser registrados.
Precondiciones:	Ninguna.
Flujo Normal:	1. El actor envía los datos de la matricula del alumno para su registro. 2. Retorna un mensaje de confirmación del registro.
Flujo Alternativo	2'. En caso de que ocurra algún error en el registro del alumno retorna un mensaje de error.

PostCondiciones Ninguna.

Nombre:	Modificar alumno
Actores:	Sistema Académico, Sistema de Tesorería
Descripción:	El actor envía los datos ya sea del alumno, el apoderado y la matricula para ser modificados.
Precondiciones:	El sistema académico puede modificar cualquier dato personal y académico del alumno. El sistema de tesorería modifica estados de matricula, y datos con referencia a los pagos del alumno.
Flujo Normal:	1. El actor envía los datos a modificarse del alumno. 2. Retorna un mensaje de confirmación de la modificación.
Flujo Alternativo	2'. En caso de que ocurra algún error en la modificación retorna un mensaje de error.
PostCondiciones	Ninguna.

Nombre:	Consultar alumno
Actores:	Sistema Académico, Sistema de Tesorería, Sistema de Contabilidad, Sistema de Matrícula.
Descripción:	El actor envía los datos de identificación de un alumno para obtener sus datos.
Precondiciones:	Ninguno.

Flujo Normal: 1. El actor envía los datos de identificación del alumno del cual necesita consultar sus datos. 2. Se le retorna al actor los datos solicitados.
Flujo Alternativo 2'. En caso de que los datos de identificación no sean válidos se le envía un mensaje de error.
PostCondiciones Si se solicitan los datos del apoderado se puede hacer la consulta de los datos del apoderado.

Nombre:	Consultar apoderado
Actores:	Sistema Académico, Sistema de Tesorería, Sistema de Contabilidad, Sistema de Matrícula.
Descripción:	El actor envía los datos de identificación del apoderado para obtener sus datos.
Precondiciones:	Ninguno.
Flujo Normal:	1. El actor envía los datos de identificación del apoderado del cual necesita consultar sus datos. 2. Se le retorna al actor los datos solicitados.
Flujo Alternativo	2'. En caso de que los datos de identificación no sean válidos se le envía un mensaje de error.
PostCondiciones	Ninguna.

Nombre:	Crear carrera
Actores:	Administrador del sistema
Descripción:	El actor desea crear una nueva carrera.
Precondiciones:	Ninguna.
Flujo Normal:	1. El actor ingresa los datos correspondientes a la nueva carrera que se desea crear. 2. Selecciona el botón de Crear.
Flujo Alternativo	2'. En caso de que ocurra algún error en la creación retorna un mensaje de error.
PostCondiciones	Ninguna.

Nombre:	Modificar carrera
Actores:	Administrador del Sistema
Descripción:	El actor desea modificar los datos de una carrera.
Precondiciones:	La carrera debe haberse creado previamente.
Flujo Normal:	1. Se selecciona la carrera de la lista correspondiente. 2. Se selecciona el botón Modificar. 3. Se ingresan los datos de la carrera que se desean modificar. 4. Se selecciona el botón Grabar.

Flujo Alternativo	
4'. Si los datos ingresados no cumplen con la validación se muestra un mensaje de error.	
PostCondiciones	
Ninguna.	

Nombre:	Eliminar carrera
Actores:	
Administrador del Sistema	
Descripción:	
El actor desea eliminar una carrera.	
Precondiciones:	
La carrera debe haberse creado previamente. Ningún alumno que estudia actualmente debe estar matriculado en la carrera a eliminar.	
Flujo Normal:	
1. Se selecciona la carrera de la lista correspondiente.	
2. Se selecciona el botón Eliminar.	
3. Se acepta la confirmación de la eliminación.	
Flujo Alternativo	
3'. Si la carrera aun tiene alumnos estudiando actualmente muestra un mensaje de error y se cancela la eliminación.	
3'. No se confirma la eliminación.	
PostCondiciones	
Ninguna.	

Nombre:	Consultar carrera
Actores:	
Sistema de Matrícula, Sistema de Tesorería, Sistema Académico	
Descripción:	
El actor desea consultar datos de una carrera.	

Precondiciones: Ninguna.
Flujo Normal: 1. El actor solicita los datos de una o varias carreras. 2. Se le retorna al actor los datos solicitados.
Flujo Alternativo 2'. Si no se encuentran los datos solicitados se envía un mensaje de error.
PostCondiciones Ninguna.

Nombre:	Consultar horario docente.
Actores:	Sistema de Asistencia
Descripción:	El actor desea consultar el horario de un docente dado.
Precondiciones:	Ninguna.
Flujo Normal:	1. El actor solicita el horario de un personal dado. 2. Se le retorna al actor los datos solicitados.
Flujo Alternativo	2'. Si no se encuentran los datos solicitados se envía un mensaje de error.
PostCondiciones	Ninguna.

Nombre:	Generar asiento contable de ingreso de pagos de alumnos
Actores:	Sistema de Tesorería
Descripción:	El actor debe registrar un asiento contable de ingreso por el pago efectuado de un alumno.
Precondiciones:	Ninguna.
Flujo Normal:	1. El actor envía los datos de los conceptos de pago correspondientes al pago del alumno. 2. De acuerdo a los conceptos de pago se crean las cuentas contables. 3. Se procede con el registro del asiento contable. 4. Se envía al actor un mensaje de confirmación del registro.
Flujo Alternativo	2'. Si ocurre algún error en el registro del asiento se envía al actor un mensaje de error.
PostCondiciones	Ninguna.

Nombre:	Generar asiento contable de devolución de dinero al alumno
Actores:	Sistema de Tesorería
Descripción:	El actor desea registrar un asiento contable de devolución de dinero a un alumno.
Precondiciones:	Ninguna.

Flujo Normal: 1. El actor envía los datos de los conceptos de pago correspondientes a la devolución de dinero del alumno. 2. De acuerdo a los conceptos de pago se crean las cuentas contables. 3. Se procede con el registro del asiento contable. 4. Se envía al actor un mensaje de confirmación del registro.
Flujo Alternativo 2'. Si ocurre algún error en el registro del asiento se envía al actor un mensaje de error.
PostCondiciones Ninguna.

Nombre:	Crear modalidad
Actores: Administrador del sistema	
Descripción: El actor desea crear una nueva modalidad para matrícula.	
Precondiciones: Ninguna.	
Flujo Normal: 1. El actor ingresa los datos de la nueva modalidad a crear. 2. Se selecciona el botón Crear. 3. Se muestra un mensaje de confirmación del registro exitoso.	
Flujo Alternativo 2'. Si no cumple con la validación de los datos se muestra un mensaje de error y no se procede con el registro.	
PostCondiciones Ninguna.	

Nombre:	Modificar fecha de validez de modalidad
Actores:	Administrador del sistema
Descripción:	El actor desea modificar una modalidad para matrícula.
Precondiciones:	La modalidad debe estar creada previamente. Solo puede modificarse la fecha de validez de una modalidad.
Flujo Normal:	1. El actor selecciona la modalidad que desea modificar de una lista. 2. Se selecciona el botón Modificar. 3. Se ingresa una nueva fecha de validez. 4. Se selecciona el botón Grabar.
Flujo Alternativo	4'. Si existe algún matriculado en esa modalidad y tiene fecha de matrícula mayor a la nueva fecha de validez se muestra un mensaje de error.
PostCondiciones	Ninguna.

Nombre:	Eliminar modalidad
Actores:	Administrador del sistema
Descripción:	El actor desea eliminar una modalidad.
Precondiciones:	La modalidad debe haber sido creada previamente. No debe existir ningún alumno matriculado en esa modalidad.
Flujo Normal:	1. El actor selecciona la modalidad que desea eliminar de una lista. 2. Se selecciona el botón Eliminar. 3. Se comprueba que no exista algún matriculado en la modalidad y se solicita la confirmación de la eliminación.

Flujo Alternativo	
4'.Si existe algún matriculado en esa modalidad se muestra un mensaje de error.	
PostCondiciones	
Ninguna.	

Nombre:	Consultar modalidad
Actores:	
Sistema de Matrícula, Sistema de Tesorería	
Descripción:	
El actor desea consultar datos sobre las modalidades.	
Precondiciones:	
Ninguna.	
Flujo Normal:	
1. El actor solicita información sobre una modalidad específica o sobre todas las modalidades.	
2. Dependiendo de lo solicitado se envía al actor la información correspondiente.	
Flujo Alternativo	
2'.Si los datos de solicitud no corresponde a ninguna modalidad se envía un mensaje de error.	
PostCondiciones	
Ninguna.	

Nombre:	Consolidar asistencia docente
Actores:	
Sistema de Contabilidad	
Descripción:	
El actor desea la consolidación de la asistencia de los docentes para generar las planillas.	

Precondiciones: Ninguna.
Flujo Normal: 1. El actor solicita la consolidación de la asistencia de los docentes que han tenido registros en un periodo dado. 2. Se envía por cada docente su número de tardanzas y el numero de horas dictadas.
Flujo Alternativo Ninguno.
PostCondiciones Los registros consolidados son marcados para que no sean tomados en cuenta en otras planillas.

Nombre:	Consolidar asistencia administrativos
Actores:	Sistema de Contabilidad
Descripción:	El actor desea la consolidación de la asistencia del personal administrativo para generar las planillas.
Precondiciones:	Ninguna.
Flujo Normal:	1. El actor solicita la consolidación de la asistencia del personal administrativo que ha tenido registros en un periodo dado. 2. Se envía por cada trabajador, su número de días laborados, faltas y tardanzas injustificadas.
Flujo Alternativo	Ninguno.
PostCondiciones	Los registros consolidados son marcados para que no sean tomados en cuenta en otras planillas.

Nombre:	Consolidar asistencia personal externo
Actores:	Sistema de Contabilidad
Descripción:	El actor desea la consolidación de la asistencia del personal externo para generar las planillas.
Precondiciones:	Ninguna.
Flujo Normal:	1. El actor solicita la consolidación la asistencia del personal externo que ha tenido registros en un periodo dado. 2. Se envía al actor la información solicitada.
Flujo Alternativo	Ninguno.
PostCondiciones	Los registros consolidados son marcados para que no sean tomados en cuenta en otras planillas.