Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

MODELO BASE PARA INTERCONEXIÓN DE SISTEMAS INFORMÁTICOS BIBLIOTECARIOS HETEROGÉNEOS MEDIANTE LA APLICACIÓN DE UNA ARQUITECTURA ORIENTADA A SERVICIOS

Rodolfo Antonio Zea Posadas

Asesorado por el Ingeniero Roberto Sánchez de León

Guatemala, noviembre de 2013

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

MODELO BASE PARA INTERCONEXIÓN DE SISTEMAS INFORMÁTICOS BIBLIOTECARIOS HETEROGÉNEOS MEDIANTE LA APLICACIÓN DE UNA ARQUITECTURA ORIENTADA A SERVICIOS

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

RODOLFO ANTONIO ZEA POSADAS

ASESORADO POR EL ING. ROBERTO SÁNCHEZ DE LEÓN

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO EN CIENCIAS Y SISTEMAS

GUATEMALA, NOVIEMBRE DE 2013

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANO	Ing. Murphy Olympo Paiz Recinos
VOCAL I	Ing. Alfredo Enrique Beber Aceituno
VOCAL II	Ing. Pedro Antonio Aguilar Polanco
VOCAL III	Inga. Elvia Miriam Ruballos Samayoa
VOCAL IV	Br. Walter Rafael Véliz Muñoz
VOCAL V	Br. Sergio Alejandro Donis Soto
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANO	Ing. Murphy Olympo Paiz Recinos
EXAMINADOR	Ing. César Augusto Fernández Cáceres
EXAMINADOR	Ing. Ludwing Federico Altán Sac
EXAMINADOR	Ing. César Rolando Batz Saquimux
SECRETARIA	Inga. Marcia Ivónne Véliz Vargas

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

MODELO BASE PARA INTERCONEXIÓN DE SISTEMAS INFORMÁTICOS BIBLIOTECARIOS HETEROGÉNEOS MEDIANTE LA APLICACIÓN DE UNA ARQUITECTURA ORIENTADA A SERVICIOS

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería en Ciencias y Sistemas, con fecha 26 de septiembre de 2012.

Rodolfo Antonio Zea Posadas

Guatemala, 1 de abril de 2013

Ingeniero
Marlon Pérez Turk
Director de Escuela
Escuela de ingeniería en ciencias y sistemas
Facultad de Ingeniería, USAC

Señor director:

Atentamente me dirijo a usted para informarle que he tenido a bien asesorar el trabajo de tesis: **"MODELO BASE PARA INTERCONEXIÓN DE SISTEMAS INFORMÁTICOS BIBLIOTECARIOS HETEROGÉNEOS MEDIANTE LA APLICACIÓN DE UNA ARQUITECTURA ORIENTADA A SERVICIOS"**, realizado por el estudiante **Rodolfo Antonio Zea Posadas**, quien se identifica con carné **No. 2003-13211;** previo a optar el título de Ingeniero en Ciencias y Sistemas.

Al respecto quiero indicarle que luego de efectuadas las revisiones y correcciones del caso, encuentro satisfactorio el trabajo por lo que procedo a aprobarlo y remitirlo a usted para su trámite correspondiente.

Atentamente,

Ing. ROBERTO SÁNCHEZ DE LEÓN
Ciencias y Sistemas
Colegiado # 6651

Ing. Roberto Sánchez de León
Asesor

Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

Guatemala, 17 de Abril de 2013

Ingeniero
Marlon Antonio Pérez Turk
Director de la Escuela de Ingeniería
En Ciencias y Sistemas

Respetable Ingeniero Pérez:

Por este medio hago de su conocimiento que he revisado el trabajo de graduación del estudiante **RODOLFO ANTONIO ZEA POSADAS** carné **200313211,** titulado: **"MODELO BASE PARA INTERCONEXIÓN DE SISTEMAS INFORMÁTICOS BIBLIOTECARIOS HETEROGÉNEOS MEDIANTE LA APLICACIÓN DE UNA ARQUITECTURA ORIENTADA A SERVICIOS",** y a mi criterio el mismo cumple con los objetivos propuestos para su desarrollo, según el protocolo.

Al agradecer su atención a la presente, aprovecho la oportunidad para suscribirme,

Atentamente,

Ing. Carlos Alfredo Azurdia
Coordinador de Privados
y Revisión de Trabajos de Graduación

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA EN CIENCIAS Y SISTEMAS
COORDINACIÓN TESIS

ESCUELA DE CIENCIAS Y SISTEMAS

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERÌA
ESCUELA DE CIENCIAS Y SISTEMAS
TEL: 24767644

El Director de la Escuela de Ingeniería en Ciencias y Sistemas de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer el dictamen del asesor con el visto bueno del revisor y del Licenciado en Letras, del trabajo de graduación ***"MODELO BASE PARA INTERCONEXIÓN DE SISTEMAS INFORMÁTICOS BIBLIOTECARIOS HETEROGÉNEOS MEDIANTE LA APLICACIÓN DE UNA ARQUITECTURA ORIENTADA A SERVICIOS",*** *realizado por el estudiante RODOLFO ANTONIO ZEA POSADAS, aprueba el presente trabajo y solicita la autorización del mismo.*

"ID Y ENSEÑAD A TODOS"

Ing. Marlon Antonio Pérez Türk
Director, Escuela de Ingeniería en Ciencias y Sistemas

Guatemala, 07 de noviembre 2013

Universidad de San Carlos
de Guatemala

Facultad de Ingeniería
Decanato

Ref.DTG.800.2013

El Decano de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ciencias y Sistemas, al trabajo de graduación titulado: **MODELO BASE PARA INTERCONEXIÓN DE SISTEMAS INFORMÁTICOS BIBLIOTECARIOS HETEROGÉNEOS MEDIANTE LA APLICACIÓN DE UNA ARQUITECTURA ORIENTADA A SERVICIOS,** presentado por el estudiante universitario: **Rodolfo Antonio Zea Posadas,** procede a la autorización para la impresión del mismo.

IMPRÍMASE.

Ing. Murphy Olympo Paiz Recinos
Decano

FACULTAD DE INGENIERIA
DECANO

Guatemala, noviembre de 2013

/cc

ACTO QUE DEDICO A:

Dios	Por ser el motivo de mi existencia y el dador de todo lo que tengo.
Mis padres	Hugo Rodolfo Zea y Kyra Yojana Posadas Orellana, porque día a día no cesan de mostrarme su gran amor que me alienta a dar lo mejor de mí.
Mis hermanos	Hugo Jenner Zea Posadas, Carlos Fabricio y Evelyn Rodríguez Posadas, por todo el cariño que me han brindado, por ser parte de las bendiciones más grandes que he recibido en esta vida.
Mis amigos	Por sus muestras de cariño y amor incondicional hacia mi persona.

AGRADECIMIENTOS A:

Dios	Por ser el dador de todo lo que tengo y ser la fuente de inspiración en mi vida.
Mis padres	Hugo Rodolfo Zea y Kyra Yojana Posadas Orellana, por todo el esfuerzo que día a día realizan por encaminarme en el camino del bien. Por ese amor que demuestran con cada una de sus acciones.
Mis hermanos	Hugo Jenner Zea Posadas, Carlos Fabricio y Evelyn Rodríguez Posadas, por sus incondicionales muestras de amor.
Mis abuelos	Jenner Zea y Berta Orellana, por el cariño siempre brindado.
Mi asesor	Ing. Roberto Sánchez de León, por su excelente ayuda, consejos y paciencia hacia mi persona, por todo el cariño siempre demostrado.
Mis amigos	Por todo el amor, confianza, sinceridad y apoyo brindado. Aquellos que sin merecerlo me han regalado su amistad de manera incondicional. Difícil nombrarlos, sin embargo, ellos ya se dan por aludidos.

La Universidad de San Carlos de Guatemala	A la excelentísima casa de estudios en la que Dios me dio la oportunidad de educarme.
A mi país	A esta Guatemala hermosa en la que he tenido el privilegio de haber nacido y de haber vivido hasta el día de hoy.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

LISTA DE SÍMBOLOS

Símbolo	Significado
Gbps	Gigabits por segundo
GB	Gigabyte
GHz	Gigahercio
MB	Megabyte
MHz	Megahercio
rpm	Revoluciones por minuto

GLOSARIO

Adaptador de red	También llamado tarjeta de red es un periférico que permite la comunicación entre distintas computadoras que estén conectadas entre sí.
Adobe ® PDF	Siglas del término en inglés Portable Document Format desarrollado por la empresa Adobe Systems. Es un tipo de archivo utilizado para representar documentos de manera independiente al software, hardware o sistema operativo de la computadora.
Arquitectura cliente/servidor	Modelo de aplicación distribuida en la que todas las tareas son repartidas entre los proveedores de servicios y los clientes que son quienes consumen los servicios.
Arquitectura monolítica	Modelo de aplicación en la que todas las tareas son realizadas por el mismo computador.

Base de datos	Conjunto de datos pertenecientes a un mismo contexto y almacenados de manera estructurada para su posterior uso.
Base de datos relacional	Base de datos que basa su diseño en interconectar los distintos datos almacenados (a través de tablas) mediante relaciones.
Broker	Servicio web cuya función es registrar información de otros servicios web, la disponibilidad de los mismos y cómo podrían utilizarse. Funciona como un puente entre un cliente y un servicio web.
CD	Disco compacto.
Color refresh rate	Es el número de marcos por segundo que un monitor o pantalla puede mostrar. La unidad de medida es el hercio.
Componente de software	Elemento de un sistema de software que ofrece un conjunto de funcionalidades a través de interfaces definidas.

CPU	Unidad de Procesamiento Central. Es un dispositivo de hardware que se encuentra dentro de una computadora y que lleva a cabo todas las instrucciones de un programa de computadora. En una computadora también se le conoce como procesador.
DTO	Objeto de Transferencia de Datos es un objeto simple (sin comportamiento específico más que el almacenar datos) que se utiliza para comunicar subsistemas de software.
Diagrama de despliegue	Tipo de diagrama perteneciente al Lenguaje Unificado de Modelado (UML) que se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes.
Dirección URL	URL es un localizador de recursos uniforme, es una secuencia de caracteres que se utiliza para nombrar recursos (documentos de texto, imágenes, etc.) en internet para poder ubicarlos.

Disco duro	Dispositivo de almacenamiento de datos no volátil que emplean las computadoras y/o dispositivos electrónicos.
Dublin Core	Modelo de metadatos elaborado por la organización Dublin Core Metadata Initiative.
EBook	Libro electrónico, es decir, en formato digital que puede ser consultado a través de computadoras y/u otros dispositivos electrónicos.
Entity Framework	Es una herramienta de código abierto para mapeos entre bases de datos relacionales y objetos. Este ha sido diseñado específicamente para el .NET Framework de Microsoft.
Estación de trabajo	Computadora destinada para trabajo técnico o científico. Tiene una tarjeta de red para conectarse con otras computadoras.
Hardware	Componentes tangibles de una computadora o sistema informático.

HTML	Lenguaje de marcado de hipertexto. Hace referencia al lenguaje de marcado predominante para la elaboración de páginas web.
HTTP	Protocolo de transferencia de hipertexto, es el protocolo que se utiliza en cada transacción de internet.
HTTPS	Protocolo seguro de transferencia de hipertexto, es un protocolo que se basa en HTTP destinado a la transferencia segura de los datos de hipertexto.
In-house	Término en inglés utilizado para denotar las operaciones realizadas dentro de una compañía sin necesidad de contratar o delegar dichas tareas en personal ajeno a la misma.
Integrated development environment	Entorno de desarrollo integrado, es un programa informático compuesto por una serie de herramientas de programación. Su función es la construcción de otros programas informáticos por medio de uno o más lenguajes de programación.

Intranet	Es una red de computadoras privadas que utiliza la tecnología de internet para compartir dentro de su entorno (organización) parte de sus sistemas de información y sistemas operacionales.
ISO 2709	Estándar ISO para la descripción bibliográfica. Se denomina formato de intercambio para información bibliográfica en casetes.
IT	Tecnologías de la información.
LAN	Red de Área Local. Interconexión de una o varias computadoras con un alcance entre 200 metros y un kilómetro.
Llave primaria	En un diseño de base de datos relacional, la llave primaria es un campo o un conjunto de campos que identifica de manera única a un registro en una tabla.

MARC 21

Acrónimo de Catalogación Legible por Máquina. Es una norma para el intercambio de información que permite estructurar y a la vez identificar datos para que puedan ser reconocidos y manipulados por una computadora. Fue creado por un equipo de bibliotecarios de la Biblioteca del Congreso de los Estados Unidos de Norteamérica.

Memoria caché

Es un área especial de memoria que utiliza la unidad central de procesamiento de una computadora para reducir el tiempo de acceso a los datos que se utilizan con más frecuencia.

Memoria principal

Es la memoria de una computadora en donde se encuentra el código de instrucciones y los datos de los programas que se encuentran en ejecución.

Módulo

Es una porción de un programa con la funcionalidad de realizar una o más tareas específicas del conjunto total que lleva a cabo el programa.

Navegador de internet o navegador web	Aplicación que interpreta la información de los archivos y sitios web de internet, para que estos puedan ser leídos por una persona.
Parámetros de entrada	Datos que recibe como insumo un servicio web para poder realizar su funcionalidad.
Parámetros de salida	Datos que retorna un servicio web luego de haber concluido con su funcionalidad.
Patrón de diseño	Son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de programas informáticos. El patrón resulta ser la solución que llega a convertirse en un estándar para dicho tipo de problemas.
PDA	Del inglés Personal Digital Assistant, es una computadora de mano originalmente creada como agenda electrónica con un sistema de reconocimiento de escritura. Hoy en día también se le llama así a las computadoras de bolsillo.

Pixel	Es la menor unidad homogénea en color que forma parte de una imagen digital.
POCO	Es un acrónimo de Plain Old CLR Object utilizado por los desarrolladores de .NET de Microsoft utilizado para identificar un objeto simple.
Protocolos TCP/IP	Familia de protocolos de internet. Permiten la transmisión de datos entre computadoras.
ROI	Acrónimo de Retorno de la Inversión. Es una medida de rendimiento utilizada para evaluar la eficiencia de una o más inversiones.
Rollback	Es una operación que devuelve a la base de datos a algún estado previo.
Script	Es una secuencia de instrucciones que la computadora debe de procesar. Por lo general se almacenan en archivos de texto.

SCSI	Acrónimo de Interfaz de Sistema para Pequeñas Computadoras. Es una interfaz estándar para la transferencia de datos entre distintos dispositivos.
Servidor	Computadora en la que se ejecuta un programa que realiza alguna tarea en beneficio de otras aplicaciones (clientes).
Servidor remoto	Es un servidor que provee de servicios de información a otras computadoras. La conexión entre estos es a través de internet.
Sistema consumidor	Sistema informático que hace uso de los servicios web del *broker* u otro sistema informático.
Sistema operativo	Es un programa o un conjunto de programas que se encarga de gestionar los recursos del hardware y que adicionalmente provee servicios a los programas de aplicación.
Skin	Son estilos visuales (apariencia gráfica) que se pueden aplicar a un software o a una página web.

SOA *Compliant*	Software construido con los parámetros de diseño establecidos por la arquitectura orientada a servicios.
Software	Conjunto de los programas de cómputo, procedimientos, reglas, documentación y datos asociados, que forman parte de las operaciones de un sistema de computación.
Software propietario	Cualquier programa informático en el que el usuario tiene limitaciones para usarlo, modificarlo o redistribuirlo.
SQL	Lenguaje de consulta estructurado, es un lenguaje establecido para acceder y realizar operaciones sobre bases de datos relacionales.
SVGA	Del acrónimo Super Video Graphics Array, es un término que cubre una serie de estándares de visualización gráfica en computadoras, tarjetas de video, monitores, proyectores, entre otros.

Tabla de base de datos	Tipo de modelado de datos en donde se almacena la información que contiene la base de datos.
Tarjeta de red *ethernet*	Tarjeta de red cuya característica principal es la conexión mediante cableado. Ethernet es un estándar para redes de área local que define las características de cableado y señalización de nivel físico, así como los formatos de las tramas de datos definidos por el modelo OSI.
Tarjeta de red *wireless*	Tarjeta de red con conexión inalámbrica.
Unidad de *backup*	Dispositivo de almacenamiento destinado para realizar copias de seguridad de la información de una computadora o dispositivo electrónico.
Unidad de CD/ROM	Dispositivo utilizado para la lectura de los datos almacenados en discos compactos.
WAN	Del acrónimo red de área amplia, es un tipo de red de computadoras capaz de cubrir distancias de 100 a 1000 kilómetros.

Window-based	Característica de un programa cuya apariencia visual es a través de ventanas.
XGA	Del acrónimo en inglés Extended Graphics Array, es un estándar de visualización de gráficos para computadoras.
XML	Lenguaje de marcas extensible, es un lenguaje de marcas que permite desarrollar la gramática de lenguajes específicos de la misma manera que lo hace HTML. Ha sido propuesto como estándar para el intercambio de información estructurada entre diferentes sistemas y/o plataformas.
XSLT	Es un estándar establecido por la organización W3C que presenta una forma de transformar documentos XML en otro tipo de documentos (incluso que no son formatos XML). Es muy utilizado en la edición web para generar páginas HTML.

RESUMEN

Ante la instalación del software para bibliotecas GLIFOS en la Biblioteca Central de la Universidad de San Carlos de Guatemala y la actual iniciativa de su instalación en las distintas bibliotecas, la presente propuesta pretende aprovechar esta situación mediante el diseño de un sistema informático para facilitar la comunicación entre estos sistemas.

Teniendo como objetivo primordial: el interconectar las diversas instancias de GLIFOS, el trabajo de graduación propone el diseño de un sistema con arquitectura orientada a servicios. Este se enfocará básicamente en la comunicación, transparente al usuario final, entre los múltiples sistemas bibliotecarios, de manera que a partir de la plataforma a diseñar se pueda solicitar y retornar información a cualquiera de éstos.

Una de las características principales, del modelo a diseñar, es que deberá permitir la interconexión independientemente del lenguaje de programación y del software de administración de base de datos con que cada sistema bibliotecario externo funcione.

Con el presente trabajo, se pretende establecer un modelo base para interconexión de sistemas bibliotecarios con GLIFOS, que en un futuro pueda servir como base en el diseño de otros sistemas para otras áreas dentro de la Universidad.

OBJETIVOS

General

Definir un modelo base de interconexión de sistemas de información bibliotecarios heterogéneos, mediante la aplicación de patrones de diseño de la arquitectura orientada a servicios.

Específicos

1. Recopilar, describir y relacionar conceptos de arquitectura orientada a servicios.

2. Recopilar, describir y realizar un análisis de la situación actual del software de GLIFOS.

3. Diseñar un modelo de software base para interconectar los diferentes GLIFOS de las bibliotecas de la Universidad de San Carlos de Guatemala.

4. Definir las características, funcionalidades, requerimientos y recomendaciones a nivel de software y hardware, relacionadas con el paquete de software GLIFOS.

5. Aplicar los principios de diseño de servicios establecidos por SOA a los servicios a definirse en el modelo base.

6. Aplicar el patrón de diseño de arquitectura de software *broker forwarding*, para administrar la comunicación de los distintos servicios a interconectar.

7. Elaborar diseño arquitectónico en capas para el modelo base.

8. Elaborar diagramas esquemáticos con el modelo del sistema orientado a servicios.

9. Describir interfaz de servicios para el modelo a proponer.

10. Definir la funcionalidad completa que el modelo propone, así como la manera en que pudiese añadírsele funcionalidad para futuros escenarios.

INTRODUCCIÓN

Hoy en día la comunicación entre diversos sistemas informáticos elaborados con distintas tecnologías ya no es algo novedoso. Con el auge del internet, la necesidad de compartir información se convirtió en uno de los mayores retos para los creadores de tecnología de ese tiempo. El desarrollar un modelo para que diversos sistemas heterogéneos pudiesen comunicarse entre sí (especialmente tecnologías web), ha sido uno de los mayores logros en el ámbito de las tecnologías de la información. Con el desarrollo de modelos de comunicación, como lo es el diseño de servicios web y la creación de un patrón de diseño de software orientado a servicios, la comunicación entre sistemas de informática heterogéneos ya es posible.

Siendo este patrón de diseño válido para cualquier tecnología de desarrollo y aplicable a cualquier tipo de sistema informático, su uso puede verse en un sin fin de lados alrededor de la web. Cuestiones tan cotidianas como ver publicaciones de Twitter en Facebook, asociar una cuenta de Gmail a una cuenta de Youtube, etc., involucran comunicación mediante servicios sin importar el lenguaje de programación en el que hayan sido desarrollados.

Tomando en consideración esta premisa y el deseo de poder realizar un aporte a la Universidad de San Carlos de Guatemala, a lo largo del presente trabajo, se enfoca en el diseño de un sistema informático, capaz de intercomunicar diferentes sistemas bibliotecarios. Este podría llegar a ponerse en práctica al momento de que las distintas bibliotecas de la Universidad incorporasen sus catálogos a un sistema informático. De tal manera que se podría consultar una publicación en cualquier biblioteca de la Universidad a

través del sistema propuesto. Este mismo diseño podría abrir puertas a otro tipo de sistemas que en un futuro pudiesen llegar a implementarse tanto en la Universidad como fuera de esta.

1. FUNDAMENTOS DE LA ARQUITECTURA ORIENTADA A SERVICIOS

Durante los últimos años, la humanidad ha atestiguado grandes avances, tanto en las ciencias de la computación como en las tecnologías de la información. Lo que un día se creyó como ficción, hoy en día es una realidad. Desde avances tan complejos como el tener todo un centro de datos virtual, hasta algo tan cotidiano como el poder pagar el recibo del servicio de electricidad desde cualquier banco. O el poder reservar una habitación en un hotel al otro lado del mundo, desde cualquier computadora con tan solo dar unos clics.

El presente capítulo se enfoca en conocer un poco la evolución en la comunicación entre computadoras, iniciando con la teoría de la información. Este fue el punto de partida que desencadenó una serie de acontecimientos, que han culminado hasta el día de hoy en conceptos tan innovadores como los propuestos por el diseño orientado a servicios. Este último es el eje principal del presente trabajo de investigación.

1.1. Teoría de la información

El propósito detrás de todo sistema de comunicación es la transmisión eficiente de información desde un punto origen, a otro punto destino. La teoría de la información proporciona una serie de conceptos que permiten manejar de manera inteligente el proceso de comunicación.

1.1.1. Historia de la teoría de la información

La teoría de la información se remonta al siglo XIX, época en la que Samuel Morse junto a Joseph Henry inventaron el telégrafo. Posteriormente Morse participó en el desarrollo del método de transmisión, hoy en día conocido como código Morse. Este código trabaja con tres combinaciones posibles: el punto (un pulso de energía), la línea (una corriente continua de energía) y la ausencia de energía. Para su definición Morse investigó los símbolos más recurrentes (en el idioma inglés) y estableció cadenas de composición simple para definirlos. Por el contrario, para los símbolos menos recurrentes estableció cadenas con mayor grado de complejidad de manera que la comunicación fuese más eficiente.

A principios del siglo XX Andrei A. Markov, contribuyó al desarrollo de la teoría de la información, mediante sus aportes en la teoría de las cadenas de símbolos en literatura. En 1927, Ralph V. Hartley propone la primera medida exacta de la información asociada a la emisión de símbolos, considerada como la precursora del bit y del lenguaje de oposición binaria. Para 1936, Alan Turing realizó el esquema de una máquina capaz de tratar información con emisión de símbolos.

Para 1948, Claude Elwood Shannon publicó un trabajo llamado: Una teoría matemática de la comunicación. En este demostró que todas las fuentes de información se pueden medir. Adicionalmente expuso la similitud existente en la unidad de medida con los canales de comunicación, determinándose con ello la velocidad de transferencia o capacidad. Además, mostró que la información se puede transmitir en un canal, sí y solo sí, la magnitud de la fuente no excede la capacidad de transmisión del canal; sentó las bases para la corrección de errores, supresión de ruidos y redundancia.

1.1.2. Elementos de la teoría de la información

La teoría de la información es una disciplina científica, cuya finalidad es encontrar la manera más sencilla, segura y rápida de codificar un mensaje. Esta se basa en la premisa de que la comunicación tiene una fuente, la cual utiliza un transmisor quién emite una señal que viaja por un canal para llegar a un receptor. Este es responsable de decodificar la información convirtiéndola en un mensaje comprensible para el destinatario.

La codificación se refiere al cifrado de mensajes para asegurar su privacidad. Esto puede realizarse mediante la transformación de voz o imagen en señales eléctricas o electromagnéticas. Un concepto fundamental, dentro de lo que es la teoría de la información, es la cantidad de información en un mensaje. Esta debe de ser un valor matemático bien definido y medible. El término cantidad de información se refiere a la probabilidad que un mensaje, dentro de un conjunto de posibles mensajes, sea recibido. En este caso, cuanto mayor es la probabilidad de recibir un mensaje, menor es el valor asignado a la cantidad de información. Por ejemplo, si se tiene la certeza que un mensaje será recibido, la cantidad de información es 0.

Los elementos de la teoría de la información son los siguientes:

- Fuente: una fuente de información es todo aquello que emite mensajes. En sí misma, es un conjunto finito de mensajes de manera que no es posible extraer más de los que ésta posea; otra forma de ver las fuentes de información, son todos aquellos elementos de los cuales se puede obtener datos e información. Por ejemplo, una persona puede transmitir sus conocimientos a otra o una computadora mediante una red puede transmitir datos a otra computadora. En ambos casos es posible observar

que tanto la persona no puede transmitir más conocimiento del que tiene. Tampoco una computadora más información de la que tiene almacenada. La fuente por su naturaleza generativa puede clasificarse en aleatoria o determinística. También por la relación de los mensajes, una fuente puede ser estructurada o no estructurada. A la teoría de la información le interesan las fuentes aleatorias y estructuradas; una fuente es aleatoria cuando no se puede predecir el próximo mensaje que emitirá. Una fuente es estructurada cuando posee un nivel de redundancia.

- Mensaje: es un conjunto de elementos informativos que el emisor envía a quién cumplirá la función de receptor. Solo a través del mensaje el fenómeno comunicativo puede generase. Para poder llevar a cabo la comunicación de manera adecuada, es sumamente fundamental que ambas partes reconozcan y comprendan el código que el mensaje establece. En este contexto el código puede ser un idioma, otra serie de símbolos, señas, gestos, etc. Un ejemplo de un código de un mensaje puede ser, un conjunto de números compuesto por 0s y 1s que se envía de una computadora a otra.

- Código: es un conjunto de símbolos que se utilizan para representar un mensaje de acuerdo a reglas y convenciones. Por ejemplo, al inicio de esta sección se presentó el código Morse. En este, los impulsos eléctricos o la ausencia de ellos tiene cierto significado a partir de convenciones establecidas con anterioridad.

- Información: es un conjunto procesado de datos los cuales forman un mensaje que cambia el conocimiento del sujeto o sistema que lo recibe. En general la información tiene una estructura interna y puede ser clasificada según varias características: significado que quiere decir qué

tan importante es para el receptor, la vigencia que indica si la información es actual o desfasada, la validez que indica si el emisor es fiable o puede proporcionar información falsa y el valor que específica que tan útil resulta para el destinatario.

1.1.3. Teoría de la información aplicada a la tecnología

La teoría de la información se encuentra ligada a la tecnología actual, por ejemplo en el desarrollo de internet. En la infraestructura física como lo es el caso del protocolo TCP/IP, el cual funciona apegado a la teoría de la información, teniendo una fuente (equipo transmisor de datos el cual ordena el mensaje según el protocolo), mensaje (la información que se envía), el canal (la infraestructura de redes asociada, fibra óptica, enrutadores, etc.), un código (pulsos eléctricos), el receptor (el equipo que recibirá el mensaje) y la información que envía. En los servicios de la capa de aplicación por ejemplo las Wiki en el cual se tiene un emisor (la persona que edita el contenido), el canal (el sitio de internet), el código (el lenguaje idiomático), el receptor (el lector que recibe la información) y la información transmitida.

Para los objetivos del presente trabajo, es importante comprender como la arquitectura orientada a servicios es una aplicación de la teoría de la información.

En este caso se tiene el emisor que sería la capa de datos del sistema, el canal seria el servicio que se consume mediante internet y el receptor sería las aplicaciones que han consumido el servicio web. De esta manera es posible observar que la teoría de la información se aplica adecuadamente a los sistemas con arquitectura orientada a servicios.

1.2. Arquitectura orientada a servicios

La arquitectura orientada a servicios define un marco de diseño para la integración de sistemas de software independientes de manera que desde la red pueda accederse a sus funcionalidades. La forma en que puede hacerse uso de dichas funcionalidades es mediante el consumo de servicios.

1.2.1. Servicios

Según la Real Academia Española, el término servicio se define como la acción y efecto de servir. Ampliando esta definición es válido decir que prestar un servicio implica estar a disposición de alguien ejecutando sus órdenes; en economía y mercadeo un servicio es un conjunto de actividades que realiza la empresa para atender las necesidades del cliente. El servicio tiene distintas características que lo diferencian entre las que se puede mencionar lo intangible (no puede verse, probarse, olerse, oírse y sentirse), antes de la compra y se está sujeto a esperar la finalización del servicio para determinar si fue satisfactorio o no. Otra característica es lo heterogéneo (dos servicios similares nunca serán iguales), lo inseparable (la producción y el consumo son parcial o totalmente simultáneos), lo perecedero (los servicios no pueden ser almacenados) y la ausencia de propiedad (los consumidores adquieren el derecho de consumir y a acceder al servicio más no la propiedad).

Desde épocas remotas, el ser humano se ha basado en el apoyo a otras personas en tareas específicas y con eso ha suministrado servicios. También, el apoyo de grupos que llevan a cabo tareas colectivas en apoyo a una tarea más grande que presta un servicio.

Un ejemplo es el servicio de entrega de comida a domicilio. Desde que se hace la llamada telefónica, una persona encargada del PBX se encarga de atender y tomar la orden. Posteriormente, un cocinero procede a preparar la comida y un repartidor la lleva hasta la dirección de entrega. En este caso es posible observar que individualmente se prestaron distintos servicios (atención de llamada, preparación de alimentos y entrega a domicilio), que en conjunto hacen un servicio que es la entrega de comida rápida a domicilio. Con ello es posible concluir que para que este servicio resulte efectivo, cada uno de los servicios individuales necesita tener características comunes, como disponibilidad, confiabilidad y comunicación en el mismo lenguaje. Cumpliendo todo eso es posible tener un servicio de entrega de comida rápida a domicilio productivo.

1.2.2. ¿Qué tipos de servicios existen?

Existen muchos tipos de servicios, pero los que se abordarán en este trabajo son aquellos utilizados por las aplicaciones SOA (Arquitectura Orientada a Servicios). Dentro de este universo existen varias clasificaciones las cuales varían según el criterio del autor al que se haga referencia; pero básicamente se tomarán tres tipos de servicios:

- Servicios de control: se encargan de recibir peticiones del cliente y realizar las peticiones necesarias a otros servicios para retornar una respuesta. En otras palabras, se encargan de coordinar a los otros servicios.

- Servicios de negocio: son aquellos servicios que implementan una tarea de negocio y forman parte del proceso de negocio. Estos servicios por lo

general son poco reutilizables ya que están orientados a resolver una tarea específica.

- Servicios de utilidad: representan una tarea altamente reutilizable. Se pueden clasificar en dos tipos: servicios orientados al negocio, que representan una tarea de negocio reutilizable y servicios tecnológicos encargados de encapsular una tecnología determinada.

Las aplicaciones SOA se pueden dividir en tres capas: capa de recepción de peticiones, capa de tareas, capa de lógica reutilizable.

1.2.3. Principios de la orientación a servicios

Una aplicación SOA debe cumplir los siguientes principios para considerarse SOA Compliant, esto según Thomas Erl:

- Reusabilidad: el servicio debe de ser diseñado y construido, pensando en su uso masivo.

- Contrato formal: todo servicio debe de proporcionar un contrato en el cual figuren el nombre del servicio, la forma de acceso, la funcionalidad que ofrece, los datos de entrada de las funciones y los datos de salida.

- Bajo acoplamiento: los servicios deben de ser independientes unos de otros.

- Composición permitida: todo servicio debe de ser construido pensando en que se construirán servicios genéricos de más alto nivel. A su vez podría estar compuesto por servicios de más bajo nivel.

- Autonomía: el servicio debe de tener su propio entorno de ejecución

- Sin estado: un servicio no debe de guardar ningún tipo de información

- Visible: todo servicio debe de poder ser visible de alguna forma para que pueda ser utilizado de modo sea posible evitar la creación accidental de servicios con las mismas funcionalidades.

- Abstracción: un servicio únicamente muestra su interfaz en términos de operaciones mientras que el detalle de su implementación permanece oculto al consumidor.

1.3. Definición de arquitectura orientada a servicios

Muchos autores han definido SOA de distintas maneras. Dependiendo del área de enfoque a la que se haga referencia así es el punto de vista de su definición. Por ejemplo para un arquitecto de software, el concepto SOA puede diferir completamente al concepto que tiene el vendedor de tecnología, quién la mira como una herramienta que ayudará al cliente a recombinar sus componentes de software según sus necesidades. Para el gerente de la empresa es una herramienta que le ayudará a maximizar el ROI (Retorno de la Inversión) e incluso del programador para quién es un paradigma de programación. Teniendo en consideración esto, se hará hincapié en algunas definiciones que pueden diferir en algunos puntos.

En general, SOA es una arquitectura de software distribuido que consiste en varios servicios autónomos, de manera que éstos pueden ejecutarse en distintos entornos de diferentes proveedores. Su objetivo fundamental es desarrollar aplicaciones, las cuales puedan estar implementadas en distintas

plataformas e inclusive en distintos lenguajes de programación. SOA provee protocolos estándares de comunicación los cuales son capaces de alojar servicios y comunicarse con otros con el fin de intercambiar información.

1.3.1. SOA como un patrón de diseño

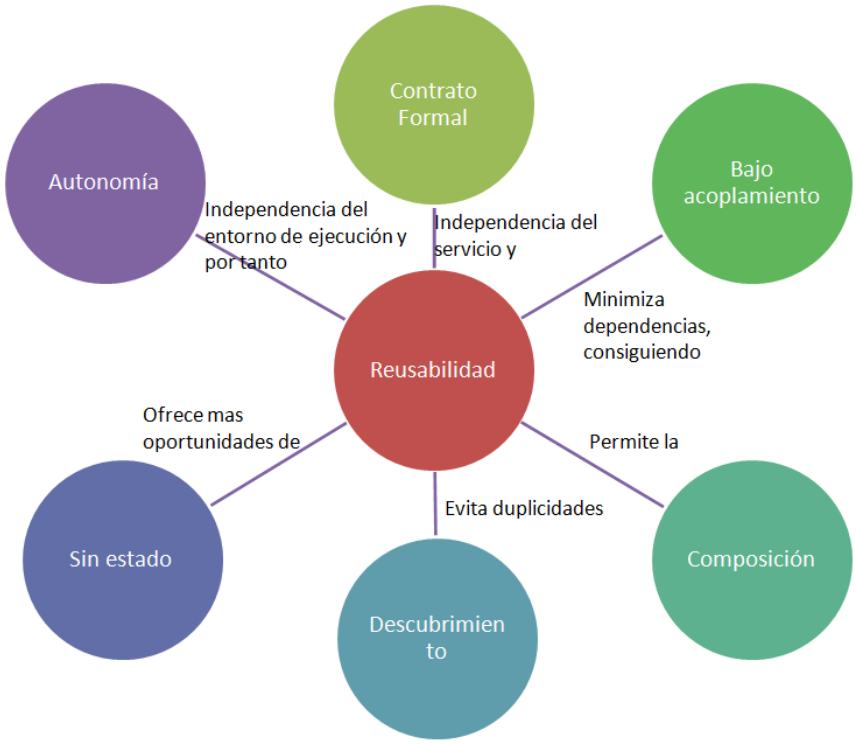
Desde hace mucho tiempo SOA, es utilizado como un patrón de diseño para el cual un servicio consumidor llama a otro servicio. En este caso, mediante la descripción del servicio, el consumidor obtiene la información necesaria del proveedor (datos de conexión y operaciones disponibles en el servicio) para ser consumido.

Un *broker* (intermediario), que es un servicio controlador, es utilizado para encontrar un servicio mediante un registro que aloja información acerca del servicio y su ubicación. Los beneficios de utilizar SOA como un patrón de diseño son: la reducción de costo del tiempo de vida de una aplicación, la capacidad de lanzamiento rápido al mercado (por la alta reutilización) y el desarrollo de aplicaciones sólidas y resistentes al cambio.

1.4. Principios de diseño de servicios

En la sección de principios de la orientación a servicios se detallan los principios claves del diseño de servicios, los que a continuación se muestran en un ámbito de una aplicación totalmente orientada a servicios.

Figura 1. **Principio de diseño SOA**



Fuente: http://2.bp.blogspot.com/-Vqdf-8xPBEI/TWxxtpw1h4I/AAAAAAAAACA/tWyuGElwNpc/s1600/soaPrinciples.png. Consulta: 20 de noviembre de 2012.

1.4.1. Estandarizando los contratos de servicios

Los distintos propósitos y capacidades de un servicio se expresan a través de un contrato y el estandarizarlo es una de las partes fundamentales de la orientación a servicios. Haciendo énfasis en cómo los servicios expresan su funcionalidad. Estos estándares definen sus tipos y modelos de datos, así como la forma en que éstos impondrán y amarrarán sus políticas. Los contratos de servicios deben ser a la vez optimizados con la granularidad apropiada y se

deben de estandarizar de manera que se garantice que los puntos de conexión son consistentes, confiables y gobernables.

1.4.2. El bajo acoplamiento en los servicios

El término acoplamiento hace referencia a una conexión o relación entre dos cosas y puede compararse al nivel de dependencia que existe entre ellas. Para este principio, se crea un tipo específico de relación dentro y fuera de los límites del servicio y se hace énfasis en reducir la dependencia entre el contrato de servicio, su implementación y cualquier consumidor del servicio.

Tener un bajo acoplamiento entre servicios, promueve mejoras significativas en la implementación de la lógica siempre y cuando exista la correcta interoperabilidad de referencia con los consumidores que han llegado a confiar en el servicio.

1.4.3. Abstracción y servicios

En este principio se enfatiza el hecho de necesitar ocultar los detalles de un servicio tanto como sea posible. De modo que para el consumidor del servicio no esté disponible su implementación.

En sí los distintos tipos de datos que describen un servicio se conocen de forma conjunta como tipos de meta abstracción. Estos son de gran importancia cuando se aplica este principio en los servicios, puesto que son los que se abstraen. Mediante su diseño es posible establecer qué información debería quedar pública y cual no. Existen cuatro tipos que son los siguientes:

- Información acerca de la calidad del servicio

- Información acerca de la tecnología
- Información funcional
- Información de la lógica programática

Se debe de regular cuánto de cada tipo se puede publicar como parte del contrato del servicio.

1.4.4. Reutilización de servicios

Es una parte esencial del análisis y diseño orientado a servicios. Cada componente o servicio puede ser utilizado en distintos ambientes cubriendo nuevas necesidades del negocio. Con la reutilización se logra reducir costos de mantenimiento, un mejor aprovechamiento de los recursos implementados y mejorar la productividad. Por consiguiente obtener el ROI (retorno sobre la inversión).

Definir, desarrollar e implementar estos servicios reutilizables es una tarea muy complicada ya que, además de utilizar herramientas específicas, es necesario utilizar estándares y buenas prácticas.

1.4.5. Autonomía en los servicios

Para que un servicio pueda ser consistente y confiable su lógica de resolución debe de tener control significativo sobre su entorno y recursos. Con ello se impulsan las características de diseño que incrementan la predictibilidad en su comportamiento.

Al implementar un servicio se debe de estudiar los niveles de aislamiento y consideraciones de normalización para lograr un nivel adecuado de autonomía. Esto aplica especialmente para servicios reutilizables que son frecuentemente compartidos.

1.4.6. Carencia de estado

La excesiva cantidad de información acerca del estado de un servicio afecta directamente su disponibilidad y disminuye su nivel de escalabilidad. Por lo tanto, los servicios son diseñados para no tener un estado a menos que esto sea totalmente indispensable.

Para la aplicación de este principio es necesario realizar una evaluación de medidas realistas que permitirán definir el nivel de carencia de estados a la que se desearía llegar regido por la arquitectura del entorno tecnológico. Dependiendo de dicho entorno se podría disponer de diversas opciones para delegar la administración de los estados de los servicios.

1.4.7. Visibilidad de los servicios

Todos los servicios deben de ser fácilmente identificados y comprendidos cuando se presenten oportunidades para su reutilización. Por ello, el servicio debe de tomar en cuenta la calidad de comunicación y su capacidad individual, a parte del mismo mecanismo que lo haría visible.

Para hacer un servicio visible, es recomendable seguir las siguientes actividades:

- Documentar la información del servicio de manera consistente

- Almacenar la información del servicio documentada en un repositorio que esté disponible para ser consultado.

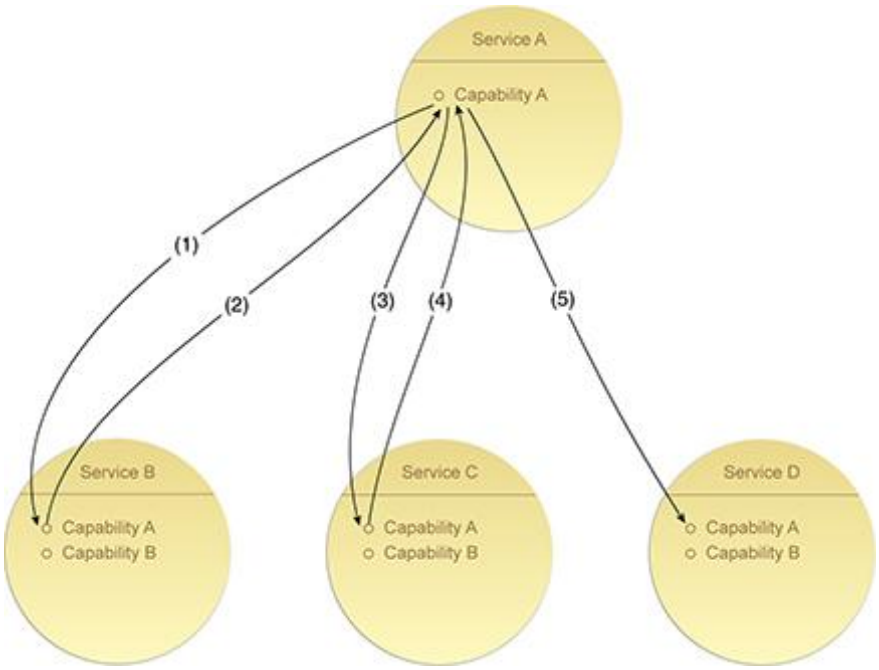
- Permitir a otras personas buscar la información documentada del servicio de una manera eficiente.

Una efectiva aplicación de este principio de diseño requiere que, la información almacenada sobre cada servicio, sea consistente y que agregue valor al entendimiento del servicio. Este principio es mejor aplicado en la fase de análisis del diseño orientado a servicios ya que durante este momento los detalles acerca del servicio, el propósito y su funcionalidad están disponibles.

1.4.8. Composición de los servicios

La composición de los servicios es un término utilizado en la arquitectura orientada a servicios, para describir la manera en que los servicios pueden enlazarse.

Figura 2. **Composición de los servicios**



Fuente: http://whatisrest.com/static/images/introduction_to_services/fig7.jpg.
Consulta: 22 de noviembre de 2012.

En la figura 2 es posible observar que un problema grande, llamado A puede subdividirse en problemas pequeños y para cada uno de estos problemas pequeños puede aplicarse unidades de lógica resolutiva: B, C, D que resuelven cada uno de los problemas pequeños; en la segunda parte del dibujo se observa como las distintas unidades de lógica resolutiva son ensambladas en una configuración que les permite resolver el problema grande A.

En este caso, las unidades resolutivas podrían ser servicios. Volviendo a configurar cada una de las unidades resolutivas para resolver otros problemas, se aumentaría el grado de reutilización y con ello se reducirán los costos y tiempos a invertirse en la solución.

Algunos factores que determinan la composición potencial de un servicio incluyen:

- Habilidad de proveer funcionalidad en diferentes niveles con distintos procesos de negocio.

- Un patrón de intercambio de mensajes

- Si el servicio soporta las características de transacciones y *rollback/compensation*.

- Soporte para manejo de excepciones

- Disponibilidad de meta datos, acerca de las capacidades y comportamiento del servicio.

1.4.9. Interoperabilidad de los servicios

La interoperabilidad es un requisito fundamental para cada uno de los principios de orientación a servicios. Por ello, la interoperabilidad no puede subsistir como un principio aislado y es tan importante que un servicio no puede existir sin ser interoperable. Todos los principios anteriores contribuyen de alguna manera al desarrollo de la interoperabilidad. Por ejemplo, los contratos de servicios garantizan una referencia a la interoperabilidad que se asocia con la coordinación de los distintos modelos de datos. Disminuir el acoplamiento hace que los servicios sean independientes y por lo tanto propensos a solucionar distintos problemas. Cuando se abstraen los detalles del servicio, permite al servicio inferior evolucionar de una forma independiente. La reutilización asocia un alto nivel de interoperabilidad entre el servicio y cualquier

consumidor potencial. Con mayor autonomía, el comportamiento del servicio se vuelve predecible en forma consistente y por lo tanto más reutilizable. Con el diseño de un servicio sin estado, la disponibilidad y escalabilidad se incrementan, permitiendo al servicio interoperar de manera frecuente y confiable. La visibilidad del servicio permite a otros servicios ubicarlo y por lo tanto facilita la interoperabilidad. Por último, para que exista composición de servicios, éstos deben de ser interoperables.

Existe un límite en la interoperabilidad de un servicio, que viene determinado por: el grado en el cual los principios de orientación a servicios han sido aplicados de forma consistente y exitosa y el nivel de la plataforma tecnológica en la cual se apoyan.

1.5. Patrones de diseño arquitectónico mediante *broker*

En un ambiente SOA, el término *broker* hace mención a los objetos que operan como intermediarios entre los consumidores y los servicios proveedores. Estos permiten al consumidor no tener que almacenar información acerca de la ubicación y él consumir el servicio proveedor. Los más novedosos proveen directorios de nombres de servicios y directorios de servicios comerciales, para que dichos servicios puedan ser localizados más fácilmente.

El *broker* debe de actuar como intermediario entre los clientes y los servicios. Estos últimos a su vez deben de estar registrados con el *broker* y los clientes deben de localizar los servicios a través de éste. Ya que se encuentra registrado, el cliente ubica al servicio por medio del *broker* y una vez se ha realizado la conexión, la comunicación cliente y servicio puede llevarse a cabo.

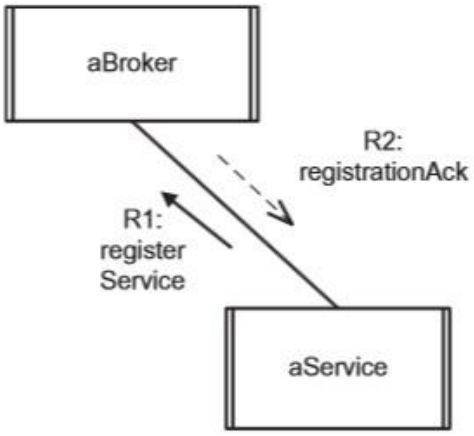
El *broker* es capaz de proveer transparencia a nivel de ubicación, así como transparencia a nivel de plataforma. La transparencia a nivel de ubicación se refiere a que si un servicio cambiara de ubicación, el cliente consumidor no lo notaría puesto que este únicamente se comunica con el *broker*. En este caso el *broker* es quién debería cambiar su configuración para comunicarse con el servicio en su nueva ubicación. La transparencia a nivel de plataforma indica que cada servicio podría ejecutarse bajo diferente tipo de hardware o software. No necesariamente se ve obligado a mantener información de la plataforma (hardware o software) en la que otros servicios se encuentran ejecutando. En este caso, debido a que los consumidores se comunican con el *broker*, éstos desconocen por completo la plataforma en la que los servicios consumidos se encuentran.

Bajo estos patrones, los clientes de los servicios no se comunican directamente con ellos, sino que mediante el *broker* hacen la petición al servicio que desean consumir y es el *broker* el que se encarga de hacer la petición a los servicios y retornar al cliente el resultado obtenido.

1.5.1. Patrón de registro de servicios

En este patrón, es necesario que el servicio registre su información (ubicación, nombre, operaciones disponibles, datos y formato de esperados, datos y formato de respuesta, etc.) al *broker*.

Figura 3. **Patrón de registro de servicios**



Fuente: HASSAAN, Gomaa. Software Modeling and Design. p. 281.

Este registro se lleva a cabo la primera vez que el servicio a ser consumido se une al *broker*. Si la ubicación del servicio cambia, es necesario realizar el registro nuevamente. En la figura 3 se ilustra este patrón y en ella puede apreciarse como el servicio envía su petición de registro (R1) al *broker* para que posteriormente lo almacene en su registro y envíe un mensaje (R2) informando al servicio el resultado del registro.

1.5.2. Patrón broker forwarding

En este patrón, el cliente envía un mensaje al *broker* con la estructura tal y como el servicio solicitado lo requiere. El *broker* identifica la ubicación del servicio hacia donde el cliente desea comunicarse y prosigue a reenviarle el mensaje recibido. Luego de obtener respuesta de parte del servicio, el *broker* la envía al cliente que originalmente realizó la petición.

Para describir mejor este patrón a continuación se detalla la secuencia de mensajes que se envían durante la comunicación cliente-*broker*-servicio:

- El cliente envía una solicitud de consumo de servicio al *broker*
- El *broker* identifica la ubicación del servicio y reenvía la petición al servicio solicitado.
- El servicio lleva a cabo la operación solicitada y posteriormente envía una respuesta al *broker*.
- El *broker* reenvía la respuesta al cliente

En resumen, es posible decir que este patrón provee un intermediario (*broker*) para cada mensaje que se envía entre el cliente y el servicio. La ventaja de este patrón es el alto grado de seguridad que provee por el hecho de que cada mensaje puede ser vetado. Sin embargo, también tiene una desventaja ya que su rendimiento es inferior al tradicional esquema de cliente-servicio, en la que la comunicación es directa.

1.5.3. Patrón broker handle

Este patrón mantiene la peculiaridad de ocultar la ubicación real de los servicios a los clientes. A diferencia de otros patrones, éste añade la ventaja de reducir el tráfico de mensajes que se envían durante la comunicación entre el cliente y el servicio. En este caso, en lugar de reenviar cada mensaje del cliente al servicio, el *broker* retorna un controlador de servicios (*service handle*) al cliente, para que sea utilizado por el cliente y comunicarse de manera directa con el servicio que desea.

Este patrón es de gran utilidad cuando se prevee una interacción prolongada mediante el envío y recepción de múltiples mensajes entre el cliente y el servicio. La secuencia de mensajes que se envían en este patrón es la siguiente:

- El cliente envía una solicitud de consumo de servicio al *broker*
- El *broker* identifica la ubicación del servicio y envía un controlador de servicios al cliente.
- El cliente utiliza el controlador para realizar la petición al servicio deseado.
- El servicio procesa la solicitud y envía directamente al cliente el mensaje de respuesta.

Este patrón, es más eficiente que el *broker forwarding*, el cliente y el servicio mantendrán una comunicación continua y de múltiples mensajes. Esto se debe a qué la comunicación entre el cliente y el *broker* solo se lleva a cabo una única vez cuando se quiere iniciar la comunicación entre el cliente y el servicio. La mayoría de *brokers* comerciales utilizan este patrón debido a que la responsabilidad de liberar los recursos del controlador, mediante el cual se comunicaron con el servicio, queda del lado del cliente.

1.5.4. Patrón descubridor de servicios

Hasta el momento todos los patrones de *broker* que se han abordado en este capítulo, en los que el cliente conoce el servicio específico al que desea comunicarse, son conocidos como *white page brokering* (*broker* de páginas

blancas). A diferencia de este tipo existe otro conocido como *yellow page brokering* (*broker* de páginas amarillas). Estos nombres hacen referencia al directorio telefónico, en el que mediante las páginas blancas se ubica con exactitud a quién se desea comunicarse; pero con las páginas amarillas es posible buscar de entre muchos servicios el que más se adecue a lo requerido.

De esta manera también funcionan los *brokers* de páginas amarillas y los *brokers* de páginas blancas. Con los *brokers* de páginas blancas, el cliente conoce con exactitud el servicio que desea consumir. Por el contrario, con los *brokers* de páginas amarillas el cliente conoce el tipo de servicios que desea consumir más sin embargo no el servicio en específico.

El patrón descubridor de servicios utiliza un *broker* de páginas amarillas debido a que es posible para el cliente el descubrir nuevos servicios. En este caso, el cliente envía una solicitud de todos los servicios de un tipo específico al *broker*. Este a su vez retorna una lista de todos los servicios disponibles pertenecientes al tipo solicitado. Luego, el cliente selecciona el servicio que se adecue a lo que requiere y envía la solicitud del servicio específico al *broker*. El *broker* retorna un cotrolador de servicios al cliente, para que este se comunique directamente con el servicio previamente solicitado.

Las distintas interacciones que se llevan a cabo con este patrón se listan a continuación:

- El cliente envía una petición de páginas amarillas al *broker* acerca de todos los servicios disponibles de un tipo específico.

- El *broker* busca la información solicitada y retorna al cliente un listado con todos los servicios disponibles dentro del tipo solicitado.

- El cliente selecciona un servicio y envía una petición de páginas blancas al *broker*.

- El *broker* ubica el servicio solicitado y retorna un controlador de servicios al cliente.

- Por medio del controlador, el cliente envía una petición al servicio deseado.

- El servicio procesa la solicitud y procede a enviar un mensaje de respuesta directamente al cliente.

2. EL SOFTWARE GLIFOS

La necesidad de almacenar y a la vez ordenar la información ha llevado al ser humano a crear formas para administrarla. Dentro de estas se encuentra el origen de la biblioteca. Este se remonta a más de cuatro mil años de historia. Con el paso del tiempo, el crecimiento poblacional y el surgimiento de las computadoras, las bibliotecas han evolucionado en su forma de operar.

Las computadoras han permitido clasificar de mejor manera los recursos bibliográficos que se almacenan en las bibliotecas. Gracias a esto, el ubicar los recursos y llevar el control de los préstamos de los mismos se ha facilitado de gran manera. Recientemente con el auge del internet y el surgimiento de los libros electrónicos, hoy en día es posible consultar la disponibilidad de libros físicos en una biblioteca, así como consultar material bibliográfico desde cualquier computadora que tenga acceso a internet.

Una de las herramientas que permite a las bibliotecas disponer de catálogos en línea es GLIFOS, y el presente capítulo se centra en describir lo esencial de este software.

2.1. Características principales

Existen en el mercado diversos productos de software destinados a la administración o gestión de documentos físicos. Algunos de ellos son de licencia pagada (software propietario) y otros se pueden clasificar en los denominados licencia libre. De estos solo algunos cumplen los estándares

necesarios para acompañar o automatizar los distintos procesos que se llevan a cabo en una biblioteca, de entre ellos se puede destacar el software GLIFOS.

2.1.1. Historia y desarrollo

GLIFOS fue desarrollado en sus inicios bajo el nombre de InfoLib 1.0 en 1993. Utilizaba una arquitectura cliente/servidor la cual ha demostrado al paso de los años ser segura, amigable y fácil de emplear. Además utilizaba un ambiente basado en ventanas (*window-based*) aprovechando la tecnología actual de su época. Como es de conocimiento generalizado, la tecnología avanza a pasos agigantados y rápidamente las tendencias cambian y por ello en 1997 se lanza InfoLib 3.0, el cual es rediseñado para tomar todas las ventajas ofrecidas por el internet.

El nuevo InfoLIb 3.0 se consideró como una nueva generación del producto. Este implementaba nuevas tecnologías de información, las cuales fueron evolucionando, así como los requerimientos bibliotecarios, forzando la aparición de nuevas versiones: 4.0, 5.0 y 6.0, hasta llegar a una nueva generación a la cual llamaron GLIFOS 7.0.

2.1.2. Características ofrecidas

Entre las características del software GLIFOS 7.0 se encuentra la capacidad de administrar contenidos digitales como documentos en formato Adobe ® PDF, *eBooks*, contenido de educación a distancia, etc. e integrarlo con contenido en material físico como es el caso de libros o mapas, entre otros.

De igual forma provee la opción de catalogar recursos de internet. Para ello utiliza el intercambio de recursos bibliográficos en formatos estándar de la

industria como: MARC21 norma para el intercambio de información que permite estructurar e identificar los datos de tal forma que puedan ser reconocidos y manipulados por computadora. MARC21 fue creado por un equipo de bibliotecarios de la Biblioteca del Congreso (EEUU). Otro de los formatos estándar es Dublin Core. Este es un modelo de metadatos en cuya implementación por lo general se utiliza XML y se define en la norma ISO 15836 de 2003.

Otra característica de GLIFOS 7.0 es que aprovecha las nuevas tecnologías de la información y comunicación para ofrecer mejores servicios a los usuarios bibliotecarios, ofreciendo acceso a contenidos remotos, administrando reservaciones y listas de espera para el material físico, automatizando el aviso al usuario por medio de correo electrónico, la revisión remota del estatus, la integración remota de diferentes bibliotecas físicas, y la capacidad de copiar registros del catálogo de otras bibliotecas con sistemas basados en la web, etc.

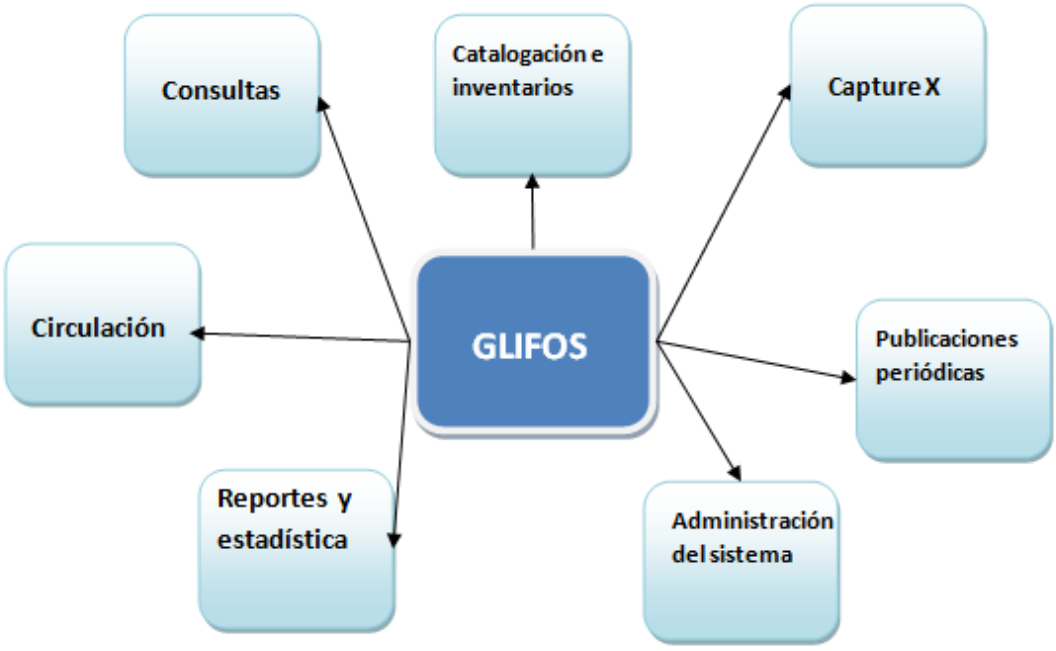
Adicionalmente provee integración de estándares actuales como XML, los cuales pueden ayudar a simplificar el intercambio de registros bibliográficos, y unido a XSLT proporciona un mecanismo para la actualización o personalización de la apariencia (*skins*) e idiomas soportados.

Según los ofrecimientos de la compañía que desarrolla este software, el licenciamiento del mismo incluye: instalación del software, documentación, capacitación y soporte técnico. La compañía no se hace responsable de la migración de datos, esto es un servicio que ofrece por separado, ya que la complejidad de realizar este proceso podría ser muy grande, requiriendo de tiempo y esfuerzo, por lo tanto es necesario examinar los elementos que estarían involucrados en la migración y determinar el costo del servicio.

2.2. Descripción de módulos

GLIFOS 7.0 está integrado por siete módulos principales, los cuales contienen la funcionalidad del sistema. Éstos se muestran en la figura 4.

Figura 4. **Módulos principales de software GLIFOS**



Fuente: elaboración propia, con programa de Microsoft Power Point.

2.2.1. Módulo de búsqueda

Provee el acceso al catálogo de GLIFOS desde cualquier computadora no importando su sistema operativo (Windows, Mac, Linux, etc.) u otro con navegador de internet que cuente con permisos de acceso al sistema. El sistema GLIFOS está construido con tecnologías que le permiten mantener

compatibilidad con la mayoría de navegadores que mantienen vigencia en el mercado al día de hoy. Por esta razón es posible tener un computador casi obsoleto, con un sistema operativo en las mismas condiciones, pero con un navegador desde Netscape Navigator 2.0 o Internet Explorer 3.0.

Las páginas de resultados, son generadas en tiempo de ejecución desde la base de datos XML en forma web con formato HTML. GLIFOS es capaz de atender a distintos usuarios en distintos idiomas, únicamente este debe haber sido previamente seleccionado.

Se provee una integración con el módulo de circulación para mostrar al usuario el estatus del material solicitado, indicando si este se encuentra actualmente o bien si se encuentra en uso. Con esto provee al usuario información sobre la lista de espera de dicho ítem. Aprovechando esta integración se permite a través de un código especial tras previa obtención, la reserva de materiales o revertir la operación.

El sistema es capaz de proveer al usuario informes estadísticos acerca de la recurrencia de préstamo sobre un artículo, agregando una sección de seguridad, por lo que la identidad de los usuarios que previamente hayan solicitado un préstamo está protegida, entregando solamente la información necesaria para que se puedan tomar las mejores decisiones al momento de realizar un préstamo bibliotecario.

El sistema es configurable para que pueda direccionar citas bibliográficas directamente a vendedores en internet. También, se pueden agregar imágenes (versión pequeña y grande) del material seleccionado. La compatibilidad se mantiene con Internet Explorer 6.0 posteriores y navegadores nuevos como Google Chrome o Mozilla Firefox.

2.2.2. Módulo de catalogación e inventario

La principal función de este módulo radica en mejorar la calidad en el proceso de ingreso de datos así como en la disminución de esfuerzo y tiempo en el catálogo e inventario. Para ello se tienen características como la utilización de una sola interfaz para el manejo del control de catalogación bibliográfica e inventario. Adicionalmente permite la configuración del formato y los campos de información bibliográfica por medio de XML y XLST para poder ajustarse a cualquier estándar de registro bibliográfico. Se integra con el módulo Capture X para la adquisición y traducción de registros bibliográficos desde el navegador.

Este módulo también tiene una conexión con el módulo de consultas de modo que antes del ingreso de un nuevo elemento se pueda verificar la existencia del mismo y con ello disminuir la duplicidad de registros. Entre otras cosas el sistema GLIFOS eliminó la creación de múltiples ventanas pensando en el usuario y evitar que este se encuentre orientado en un único punto de ingreso de información.

El sistema permite el concepto de herencia en las citas bibliográficas, de modo que con un patrón se pueden crear nuevas citas a las cuales únicamente se deban cambiar elementos distintivos como por ejemplo el tomo o número de página. También permite que se generen las citas bibliográficas en formato HTML a partir del XML, esto para ser accedidas fácilmente desde el navegador.

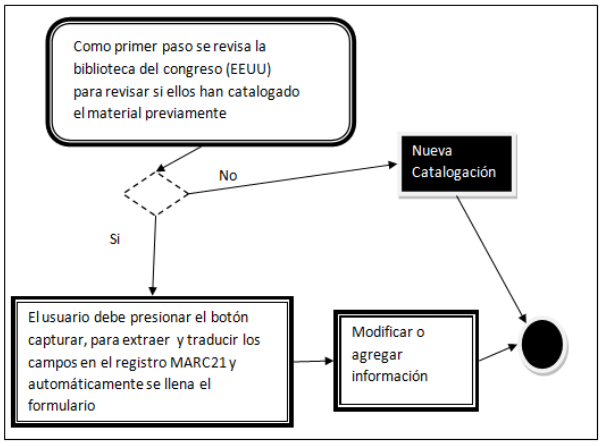
2.2.3. Módulo Capture X

Por medio de este módulo, el sistema GLIFOS es capaz de capturar, traducir e integrar registros de otras bibliotecas directamente desde sus sitios web. Los formatos bibliográficos que GLIFOS es capaz de manejar son los siguientes:

- ISO 2709
- MARC 21
- XML-MARC
- Dublin Core
- XML
- HTML

El proceso de *copy-cataloging* se ejecuta de la siguiente manera, ver figura 5.

Figura 5. **Proceso de *copy-cataloging* de GLIFOS**



Fuente: elaboración propia, con programa de Microsoft Power Point.

A través del módulo Capture X es posible importar información procedente de por lo menos cuatrocientas bibliotecas que tienen información disponible en la web y que cumplen con los estándares anteriormente mencionados. Si al momento de la administración se necesita agregar bibliotecas no incluidas se puede realizar por medio de configuraciones XLST.

Este módulo ofrece también la funcionalidad de traducción entre distintos idiomas, sobresaliendo entre estos el idioma inglés el cual es más utilizado alrededor del mundo y del que se tiene información de mayor número de registros publicados. Dependiendo de la configuración que se tenga, se traducen automáticamente de idioma la mayoría de los campos, dejando a discreción del usuario la traducción de los campos de: autor, título y editorial que usualmente se dejan en el idioma original. Esta funcionalidad es totalmente adaptable y se deja a discreción la configuración de los diccionarios de traducción, de modo que estos se adapten a las propias normas y políticas de catalogación.

2.2.4. Módulo de circulación

Este módulo se integra con los módulos de consulta, catalogación e inventario, reportes y estadísticas y consulta en línea. De esta manera si algún usuario genera un préstamo, esta transacción es procesada por el módulo de catalogación e inventario y automáticamente es reflejada por la consulta en el catálogo y por los reportes en línea.

Se tiene integrada la funcionalidad de reconocer código de barras de modo que los carné de usuarios puedan ser identificados de forma automática y rápida así como para controlar los préstamos de materiales como la devolución. El sistema está diseñado de tal forma que el personal de circulación pueda

verificar o actualizar los datos del usuario desde una única pantalla. Esto incluye validar su estado bibliotecario en cuanto a préstamos, reservas y multas y operar préstamos, reservas, cancelaciones de reservas, devoluciones, renovaciones y el respectivo pago de multas, para evitar confusiones y pérdidas de orientación al momento de operar el sistema.

El módulo de circulación incluye una administración de préstamo y reserva para distintos perfiles de tal forma que cada uno de estos tiene diferentes parámetros en cuanto al tipo de préstamo o tiempo por el cual pueden prestar un artículo. Por ejemplo mientras que una persona del personal de circulación pudiese autorizar un préstamo por quince días, el director de la biblioteca podría autorizarlo por treinta. De igual forma los materiales tienen distintos perfiles de modo que resultará distinto realizar un préstamo de libro el cual puede ser por quince días por ejemplo o el de un material en CD el cual puede ser por cinco días.

El módulo de circulación al estar integrado con los módulos de consulta, catalogación e inventario, reportes y estadísticas y consulta en línea, puede indicar al usuario si un material está disponible o no y cuan larga es la fila de espera. También el mismo usuario, tras previa autenticación y con los permisos necesarios, puede observar su estado bibliotecario. De reservar algún material en caso de decidirlo, el personal puede realizar anotaciones para que los usuarios los vean, y el propio sistema puede enviar correos electrónicos con información sobre multas o vencimientos de préstamo. Así como avisar a quien haya hecho una reserva que el libro fue devuelto. Adicionalmente el mismo sistema es capaz de calcular multas de forma detallada automatizando y transparentando el proceso.

2.2.5. Módulo de reportes y estadística

Este módulo es de gran importancia para cualquier institución pública o privada ya que a partir de la información obtenida puede ser convertida en conocimiento que puede ser aplicado de la mejor manera para mejorar la atención al usuario. Y en caso de ser una institución privada aumentar las utilidades. Para este propósito se ofrecen informes en forma impresa o en formato web. Entre otras cosas ofrece:

- Una arquitectura abierta de tecnología SQL, XML y XSLT de modo que el usuario puede diseñar informes personalizados y a la medida.

- Los reportes pueden imprimirse, generarse en formato HTML o ser enviados por correo electrónico según la conveniencia del usuario.

- Para reportes con formato HTML, para ser visibles desde un navegador web se tienen vínculos hacia otros reportes o información, por ejemplo al generar un reporte sobre algún material se puede vincular la cita completa del material.

- Informes predeterminados que por experiencias anteriores se han determinado como los más utilizados.

- Conexión con otros módulos para generación de estadística o informes

Al momento de realizar informes personalizados se deben tener conocimientos de SQL, por medio de este lenguaje se consultan las bases de datos y se obtienen y por medio de XSLT se configura la vista o forma de presentación de los datos.

2.2.6. Módulo de publicaciones periódicas

Existen publicaciones las cuales se generan de forma periódica y son una parte importante de cualquier biblioteca. Estas pueden ser revistas, periódicos o boletines, los cuales por lo general tienen sus propios estándares y normas de almacenaje y registros. GLIFOS integra un módulo que ofrece:

- Control de ejemplares de distintas publicaciones
- Control de los proveedores y/o subscriptores
- Se puede llevar el control de vencimiento y renovación de subscripciones, de tal forma que nunca se deje de recibir alguna publicación por razones de descuido en este rubro.
- Se puede catalogar artículos específicos dentro de las publicaciones
- Este módulo está integrado con el módulo de consultas, por lo que el usuario tiene tanto información de publicaciones normales, como de publicaciones periódicas.
- Informe sobre publicación periódica por título, proveedor y fecha de vencimiento.

2.2.7. Módulo de administración del sistema

Este módulo provee al usuario una forma fácil y sencilla de administrar el sistema, sin necesidad de manipular código o base de datos. Y a través de una interface controlada cuyas funciones básicas son las siguientes.

- Administración de códigos de acceso
- Administración de perfiles y permisos
- Reconstrucción e indexación automática de citas bibliográficas
- Importación y exportación de registros bibliográficos
- Edición del calendario de modo que se puedan establecer los días hábiles y los asuetos o feriados de la biblioteca, ésto es para el correcto funcionamiento de los distintos módulos que utilizan fechas para sus cálculos internos.

2.3. Arquitectura del sistema

GLIFOS inicia en una arquitectura cliente/servidor como se menciona en la sección 2.1.1. Esta arquitectura presentó en su momento ventajas significativas sobre las aplicaciones con arquitectura monolítica, bajo la cual se construían la mayoría de sistemas de información en estos años. Entre las mejoras se encuentra el poder ejecutar secuencias SQL de manera remota, concurrencia de usuarios, servicios de presentación de datos, negocios y persistencia de forma distribuida, etc.

Esta arquitectura evolucionó a través del tiempo, en arquitectura cliente-servidor mejorado, la cual desembocó en un aumento en cuanto el rendimiento. Esta arquitectura a su vez evolucionó en la arquitectura de tres capas, en la cual es posible encontrar mejoras como, la escalabilidad y flexibilidad entre otras. Luego esta arquitectura evoluciona en arquitecturas con n-capas lo cual potencia las características principales de la arquitectura de tres capas, para desembocar en arquitecturas SOA (orientadas a servicio) la cual se explica en el primer capítulo.

En general el avance tecnológico ha obligado a GLIFOS a ser acompañante de la evolución de las arquitecturas, y crecer con ellas hasta llegar al punto primordial de este trabajo, el cual se enfoca en desarrollar un componente de software, basado en arquitectura SOA compatible.

GLIFOS está orientado al funcionamiento sobre la red que puede ser LAN, WAN, Intranet o internet. De modo que los usuarios puedan tener acceso al sistema desde el campus bibliotecario, desde la red institucional, desde su casa o desde cualquier lugar que la administración lo decida, acoplándose a distintos tamaños de institución y permitiendo que el sistema de software pueda ir acoplándose al crecimiento de la infraestructura física.

2.3.1. Módulo de administración del sistema

El funcionamiento del sistema se puede recrear en distintos escenarios de uso, de acuerdo a los recursos de la institución, tamaño de la colección del material físico y otros.

2.3.1.1. Única computadora personal

La arquitectura permite la instalación de una única estación de trabajo sin necesidad de red, en la cual cliente y servidor se encuentran en la misma, aportando funcionalidad total, pensando en pequeñas instituciones que se encuentren en fase de iniciación de colecciones.

2.3.1.2. Red local

En este escenario se tienen varias computadoras conectadas por medio de una red de área local, inclusive si no tienen un acceso constante a internet. Se puede configurar el sistema de tal forma que una estación de trabajo pueda ser servidor con más funciones. De igual forma los clientes se pueden configurar, para que tengan distintas funciones como por ejemplo servicio de catálogo y búsqueda al mismo tiempo o cualquier otra combinación.

2.3.1.3. Acceso de red institucional

En este ambiente, el catálogo puede ser accedido desde cualquier parte de la institución, sin necesidad de estar físicamente en la biblioteca para consultar información sobre algún material en específico. También, se puede dar acceso a los usuarios a documentos digitales, de modo que cualquier computadora en la red por medio de un navegador pueda convertirse en cliente del sistema.

2.3.1.4. Acceso de red a través de internet

En este escenario, los usuarios el sistema tiene una conexión permanente con internet y por eso un usuario puede tener acceso al sistema desde cualquier lugar con acceso a internet y un navegador web soportado.

2.3.2. Costo de mantenimiento

El sistema ofrece un bajo costo de mantenimiento ya que se tiene distintos factores entre los que se mencionan:

- Un nivel básico de herramientas técnicas para los empleados administradores del servidor, el cual puede ser desde las versiones de Windows NT o server 2000, de modo que no es necesario que la institución posea el personal de tecnología apropiado.

- El impacto de agregar estaciones de trabajo es muy bajo ya que únicamente se necesita un navegador web considerando que según la web oficial de los propietarios del sistema, éstos no cobran licenciamiento por estación. En este caso únicamente se tendría el costo de las estaciones adicionales, así como el software necesario que sea ajeno a GLIFOS.

- Fácil administración de estaciones de trabajo, ya que no es necesario instalar ningún sistema cliente. Todo se realiza con una interfaz web y las versiones más recientes son automáticamente transferidas desde el servidor.

- Una única plataforma conlleva a que el personal ahorre recursos como tiempo ya que deberá realizar mantenimiento una única vez.

2.3.3. Requerimientos de hardware y software

Para el correcto funcionamiento del sistema GLIFOS, las especificaciones de hardware y software que deben poseer las estaciones de trabajo son las siguientes:

2.3.3.1. Servidor

- CPU: Procesador Intel de 2 GHz o superior
- Memoria principal: 1 GB.
- Disco duro: 1 disco SCSI (de preferencia 2 discos SCSI de 10,000+ rpm) con al menos 80 GB de espacio de almacenamiento disponible.
- Tarjeta de red: *Ethernet*, compatible con la red de la institución
- Unidad de *backup*: (puede ser una unidad de *backup* a través la red)
- Otros: Unidad de CD/ROM
- Sistema operativo: Microsoft NT Server Ver. 4.0 Option Pack 4 Service Pack 6a ó Microsoft Windows 2000 Server Service Pack 3 ó Microsoft Windows 2003 Server.

- Software adicional: Microsoft Internet Explorer versión 6.0 o más reciente (sin costo), Microsoft XML versión 4.0 (sin costo).

2.3.3.2. Servidor remoto

- CPU: procesador Intel de 1.5 GHz.
- Memoria principal: 1 GB o superior
- Disco duro: 100 GB de espacio libre
- Tarjeta de red: *Ethernet*, compatible con la red de la institución
- Unidad de *backup*: (puede ser una unidad de *backup* a través la red)
- Otros: Unidad de CD/ROM
- Sistema operativo: Microsoft Windows 7, Microsoft Windows 8, Microsoft Windows Server 2008 o más reciente.
- Software adicional: Microsoft Internet Explorer versión 6.0 o más reciente (sin costo), Microsoft XML versión 4.0 o más reciente (sin costo).

2.3.3.3. Estaciones de trabajo

- CPU: procesador Pentium de 1.5 GHz.
- Memoria principal: 1 GB o superior

- Tarjeta de red: *Ethernet*, compatible con la red de la institución

- Monitor: 15" con resolución SVGA (800x600 pixeles) ó XGA (1024x768) 16-bit, color *refresh rate*: 70MHz.

- Sistema operativo: Microsoft Windows 98, Windows NT, Windows 2000 ó Windows XP, Windows 7 o Windows 8.

- Software adicional: Microsoft Internet Explorer, Microsoft XML 4.0 (sin costo).

2.3.3.4. Estaciones de consulta

- CPU: procesador Intel Pentium de 1.5 GHz.

- Memoria principal: 1 GB o superior

- Tarjeta de red: *Ethernet* o *Wireless*, compatible con la red de la institución.

- Monitor: Resolución SVGA (800x600 pixeles)

- Sistema operativo: Microsoft Windows 98, Windows NT, Windows 2000, Windows XP, Windows Vista, Windows 7, Windows 8, Unix o Linux.

- Software adicional: un navegador de internet que puede ser Microsoft Internet Explorer (sin costo), Netscape Navigator (sin costo), Opera (sin costo), Mozilla Firefox (sin costo) o Google Chrome (sin costo).

2.3.3.5. PDA's de consulta

- Memoria principal: 32 *Megabytes* o superior
- Tarjeta de red: *Wireless*, compatible con la red de la institución
- Pantalla: resolución mínima de 320x240 pixeles

3. DISEÑO DEL MODELO BASE

3.1. Caso de estudio

A lo largo del presente capítulo se propone el modelo base en el cual se aplican los conceptos de una arquitectura orientada a servicios.

3.1.1. Antecedentes

La Biblioteca Central es la dependencia técnica y de servicios de la Universidad de San Carlos de Guatemala, encargada de seleccionar, adquirir, catalogar, clasificar, actualizar y mantener la conformación de un fondo bibliográfico acorde a las necesidades de los planes, programas y proyectos académicos de la Universidad[1]. Esta cuenta con una instalación del software bibliotecario GLIFOS en su servidor central. A través de éste cualquier persona mediante un navegador web y una conexión a internet puede realizar consultas sobre la existencia en el catálogo de la biblioteca. Incluso puede revisar los documentos que se tienen en formato digital en su base de datos.

A la fecha en que se realizó este trabajo de tesis, esto únicamente es posible de realizar en la Biblioteca Central. Debido a que cada una de las distintas facultades de la Universidad es independiente de las demás (cada una administra sus propios recursos), un proceso de modernización en las distintas bibliotecas no sucedería al mismo tiempo. A consecuencia de esto puede llegarse a tener una diversidad de sistemas bibliotecarios y en el mejor de los

[1] Definición que otorga la Biblioteca Central de la Universidad de San Carlos en su sitio de internet con respecto a sus funciones dentro de la Universidad. Referencia: http://www.biblioteca.usac.edu.gt/biblioteca2/QuienesSomos.html

casos se tendría el mismo software GLIFOS pero con instancias completamente diferentes e independientes unas de otras.

Considerando esta situación, se plantea el diseño de un sistema *broker* para facilitar la integración de distintas instancias de sistemas bibliotecarios. En la siguiente sección se detallan las características del *broker* que se propone.

3.1.2. Características

El sistema *broker* para cumplir con su funcionalidad debe de contar con una serie de características. A continuación se detalla cada una:

- Entradas y salidas independientes del lenguaje de programación: tanto las entradas como las salidas del *broker* deben de ser de un tipo común para todos los sistemas de información. Cualquier sistema de informática (sin importar el lenguaje de programación base) debe de poder procesarlas de manera correcta.

- Rapidez: el tiempo en el que se procesa una entrada y se genera una salida debe de ser lo más corto posible. Todas las comunicaciones a otros sistemas a partir del *broker* deben de realizarse de manera asíncrona.

- Precisión: capacidad de identificar qué información se consulta y retornar acertadamente lo solicitado. Por ejemplo, si la consulta se realiza sobre el catálogo de una biblioteca en específico, el *broker* debe consultar comunicarse específicamente con dicha biblioteca y retornar la información solicitada.

- Seguridad: cada entrada al *broker* debe validarse mediante una llave de autenticación encriptada que se recibe dentro de los parámetros de entrada. Si la llave coincide se procede a procesar la solicitud de lo contrario se rechaza la petición y se retorna un código de error. Un sistema externo que no conozca la llave de autenticación no podrá hacer uso del *broker*.

- Registro de operaciones: toda consulta realizada se almacena a manera de bitácora en una base de datos interna.

- Recuperación ante errores: durante el procesamiento de las distintas peticiones, si se produce un error el *broker* debe de ser capaz de continuar y si fuese un error fatal, se debe de retornar en el mensaje de salida el detalle del error que se produjo.

- Ocultamiento a consumidores: la información sobre la ubicación y medio de consulta, hacia las bases de datos de sistemas bibliotecarios externos se encuentra oculta para los sistemas consumidores del *broker*. De esta manera se protege que la información de los sistemas bibliotecarios sea expuesta.

- Actualización de información de sistemas bibliotecarios externos transparente: en caso que la ubicación de los servicios que proveen los diferentes sistemas bibliotecarios cambien, los consumidores del *broker* no lo percibirán ni tendrán que hacer cambio alguno puesto que la actualización la realiza únicamente el *broker*.

- Alta disponibilidad: el sistema *broker* deberá de estar disponible para su uso al menos el 95 % del tiempo. Esto considerando tiempos de

mantenimiento y eventualidades que puedan ocurrir, lo que impediría una disponibilidad de 100 %.

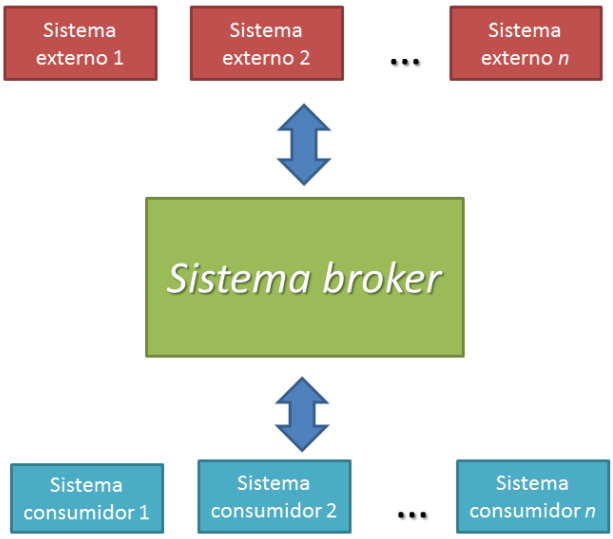
- Arquitectura por capas: es dividido en capas para tener una mejor modulación y garantizar la reutilización de componentes dentro del sistema.

- Ambiente web: el sistema dispondrá de un ambiente web administrativo (con manejo de autenticación) para consultas sobre bitácoras y reportes generados a partir de datos estadísticos sobre las consultas que ha procesado el *broker*.

3.1.3. Objetivo del sistema

El sistema a diseñar tiene como objetivo elemental el funcionar como un medio de comunicación entre sistemas de consultas y sistemas bibliotecarios externos. Para una mejor comprensión ver figura 6.

Figura 6. **Comunicación entre sistemas bibliotecarios externos y sistemas consumidores mediante *broker***



Fuente: elaboración propia, con programa de Microsoft Power Point.

En la imagen se puede ver como más de un sistema consumidor (sitio web, aplicación de escritorio, servicio o proceso del sistema operativo, etc.) puede consultar información de múltiples sistemas bibliotecarios externos (GLIFOS) a través del sistema *broker*. Este procesa las peticiones y las convierte al formato en que debe de enviarlas a los sistemas externos. Una vez recibe respuesta, nuevamente transforma el mensaje recibido en uno que pueda ser entendido por el consumidor.

Adicionalmente se busca contar con una serie de reportes web sobre datos estadísticos que provengan de la información de las consultas realizadas, tiempos de respuesta por parte de los servicios externos, errores más comunes,

sistemas externos que generan más errores, fechas y horas pico, sistemas externos más consultados, etc. Estos podrán ser consultados en un sitio web administrativo incluido dentro del sistema *broker*.

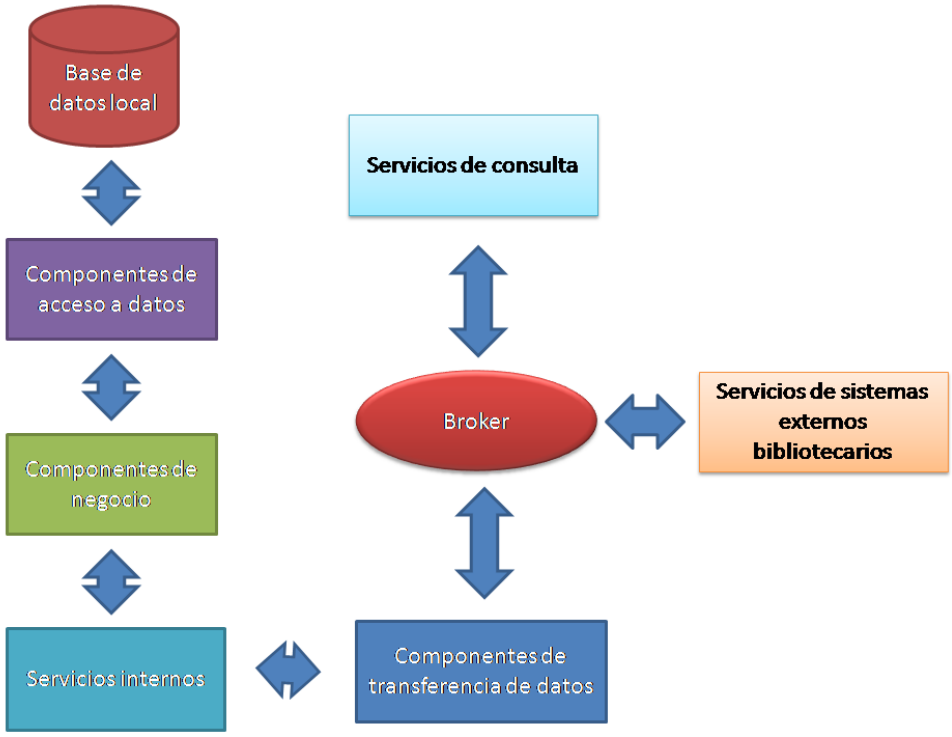
3.2. Análisis

En la fase de análisis se identifican los componentes del sistema así como las entradas y salidas para cada uno de ellos.

3.2.1. Definición de componentes

El sistema *broker* basa su diseño en una arquitectura por capas. Por lo que en cada capa se tendrá una serie de componentes reutilizables para diferentes finalidades. En la figura 7 se visualiza la manera en que los componentes se comunican entre sí. En la presente sección se clasificarán los componentes acorde a la capa del sistema en la que se encuentran.

Figura 7. **Comunicación entre componentes de sistema**



Fuente: elaboración propia, con programa de Microsoft Power Point.

- Componentes de acceso a datos: conjunto de clases que se utilizan para conectarse directamente con la base de datos. Estas clases contienen la misma estructura que la base de datos. El nombre de las clases representa el nombre de la tabla en la base de datos. El nombre de los atributos y el tipo de datos representan las columnas de la tabla. En el caso de que se trabaje con tecnologías de Microsoft se recomienda el uso de POCO (*Plain Old CLR Object*) en conjunto a *Microsoft Entity Framework* para su diseño y comunicación con la base de datos.

- Componentes del negocio: conjunto de clases que contienen la lógica de las operaciones que realiza el sistema. En estas clases es donde se hace el procesamiento de información que convierte las entradas en salidas.

- Servicios internos: puesto que se tiene una arquitectura orientada a servicios, el sistema dispone de un conjunto de servicios que comunican el *broker* con la lógica interna del sistema. Es decir, cada vez que se desea guardar un registro de una operación realizada por el *broker*, esta debe de ser enviada a través de un servicio interno. Éstos son reutilizables puesto que pueden ser llamados desde múltiples lugares en el sistema, por ejemplo desde la aplicación web administrativa y de reportes.

- Componentes de transferencia de datos: conjunto de clases que se utilizan entre la comunicación del *broker* y los servicios internos o entre la interfaz web y los servicios internos. En este caso, los servicios internos reciben objetos de transferencia de datos y responden con objetos de transferencia de datos.

- Servicios de consulta: conjunto de servicios reutilizables que se exponen a los consumidores (sistemas que desean hacer consultas a otros sistemas bibliotecarios y que utilizan el *broker* como medio de comunicación).

3.2.2. Entradas y salidas

Como todo sistema, el sistema *broker* cuenta con un conjunto de entradas y salidas. En este caso las entradas son peticiones que vienen de sistemas externos. Por otra parte, las salidas son el resultado del procesamiento de las

entradas y van en dirección del sistema *broker* hacia otros sistemas externos. Estas entradas y salidas pueden agruparse en operaciones disponibles que se detallan a continuación.

3.2.2.1. Operaciones disponibles para sistemas externos

- Consulta básica a catálogo: consiste de una búsqueda por datos generales a una, varias o todas las bibliotecas con las que se tiene conexión.
    - Entrada: cadena de caracteres en formato XML con la siguiente información:
        - Llave de seguridad
        - Identificador del sistema consumidor
        - Título
        - Autor(es)
        - Temas
        - Palabras clave
        - Tipo de material
            - ✓ Todos
            - ✓ Audio casete
            - ✓ Cuaderno
            - ✓ CD-ROM
            - ✓ Disquete
            - ✓ Folleto
            - ✓ Libro
            - ✓ Libro electrónico
            - ✓ Microficha
            - ✓ Música

        - ✓ Página web
        - ✓ Revista
        - ✓ Tesis
        - ✓ Video digital
        - ✓ Video casete VHS
        - ✓ Video casete BETA
        - ✓ DVD
      - Bibliotecas a consultar
      - Ordenar por
        - ✓ Título
        - ✓ Autor
        - ✓ Clasificación
        - ✓ Biblioteca
      - Número de coincidencias por página
      - Número de página actual
    - Salida: cadena de caracteres con formato XML con el detalle de las coincidencias encontradas. Estas se muestran bajo un listado con la siguiente información:
      - Biblioteca
      - Título de coincidencia y dirección URL para acceso directo al detalle.
      - Autor

- Consulta avanzada (búsqueda booleana): búsqueda abierta en la que es posible consultar a través de uno o más parámetros de búsqueda.
    - Entrada: cadena de caracteres en formato XML con la siguiente información:
      - Llave de seguridad
      - Identificador del sistema consumidor

        - Parámetro de entrada
            - ✓ Texto libre
            - ✓ Título
            - ✓ Autor
            - ✓ Materia
            - ✓ Editorial
            - ✓ Tipo de material
            - ✓ Clasificación
            - ✓ Serie
        - Texto a buscar para parámetro de entrada
        - Biblioteca
        - Ordenar por
            - ✓ Título
            - ✓ Autor
            - ✓ Clasificación
        - Número de coincidencias por página
        - Página actual
    - Salida: cadena de caracteres con formato XML con el detalle de las coincidencias encontradas. Estas se muestran bajo un listado con la siguiente información:
        - Biblioteca
        - Título de coincidencia y dirección URL para acceso directo al detalle.
        - Autor

- Consulta de estado en biblioteca: consulta de préstamos para un usuario específico para una biblioteca.
    - Entrada: cadena de caracteres con formato XML con la siguiente información:

        - Llave de seguridad
        - Identificador del sistema consumidor
        - Usuario
        - Contraseña
        - Biblioteca
    - Salida: cadena de caracteres con formato XML con la siguiente información:
        - Biblioteca
        - Link de acceso directo a información en biblioteca

- Consulta de material por autor: consulta de material bibliográfico por nombre de autor.
    - Entrada: cadena de caracteres con formato XML con la siguiente información:
        - Llave de seguridad
        - Identificador del sistema consumidor
        - Nombre de autor
        - Biblioteca
        - Ordenar por
            - ✓ Título
            - ✓ Autor
            - ✓ Clasificación
            - ✓ Biblioteca
        - Número de coincidencias por página
        - Número de página actual
    - Salida: cadena de caracteres con formato XML con el detalle de las coincidencias encontradas. Estas se muestran bajo un listado con la siguiente información:
        - Biblioteca

        - Título de coincidencia y dirección URL para acceso directo al detalle.
        - Autor

- Consulta de material por tema: consulta de material bibliográfico por tema.
    - Entrada: cadena de caracteres con formato XML con la siguiente información:
        - Llave de seguridad
        - Identificador del sistema consumidor
        - Tema
        - Biblioteca
        - Ordenar por
            - ✓ Título
            - ✓ Autor
            - ✓ Clasificación
            - ✓ Biblioteca
        - Número de coincidencias por página
        - Número de página actual
    - Salida: cadena de caracteres con formato XML con el detalle de las coincidencias encontradas. Estas se muestran bajo un listado con la siguiente información:
        - Biblioteca
        - Título de coincidencia y dirección URL para acceso directo al detalle.
        - Autor

- Consulta para ver detalle de coincidencia: registra la coincidencia a ver en base de datos local y navega a la URL del detalle del material bibliográfico.
    - Entrada: cadena de caracteres con formato XML con la siguiente información:
        - Llave de seguridad
        - Identificador del sistema consumidor
        - Biblioteca
        - Título de coincidencia y dirección URL para acceso directo al detalle.
        - Autor
    - Salida: navega hacia la URL de la coincidencia recibida como parámetro.

- Consulta de libros más buscados: reporte con la información de los libros más buscados.
    - Entrada: cadena de caracteres con formato XML con la siguiente información:
        - Llave de seguridad
        - Identificador del sistema consumidor
        - Número de registros por página
        - Página actual
    - Salida: cadena de caracteres con formato XML con el detalle del reporte. Estas se muestran bajo un listado con la siguiente información:
        - Biblioteca
        - Título de coincidencia y dirección URL para acceso directo al detalle.
        - Autor

- Consulta de autores más buscados: reporte con la información de los autores que han sido más consultados a través del *broker*.
    - Entrada: cadena de caracteres con formato XML con la siguiente información:
        - Llave de seguridad
        - Identificador del sistema consumidor
        - Número de registros por página
        - Página actual
    - Salida: cadena de caracteres con formato XML con el detalle del reporte. Estas se muestran bajo un listado con la siguiente información:
        - Biblioteca
        - Título de coincidencia y dirección URL para acceso directo al detalle.
        - Autor

- Consulta de temas más buscados: reporte con la información de los temas que más han sido consultados a través del *broker*.
    - Entrada: cadena de caracteres con formato XML con la siguiente información:
        - Llave de seguridad
        - Identificador de sistema consumidor
        - Número de registros por página
        - Página actual
    - Salida: cadena de caracteres con formato XML con el detalle del reporte. Estas se muestran bajo un listado con la siguiente información:
        - Biblioteca

        - Título de coincidencia y dirección URL para acceso directo al detalle.

- Consulta de tipo de material más consultado: reporte con la información de los tipos de material más consultados a través del *broker*.
    - Entrada: cadena de caracteres con formato XML con la siguiente información:
        - Llave de seguridad
        - Identificador de sistema consumidor
        - Número de registros por página
        - Página actual
    - Salida: cadena de caracteres con formato XML con el detalle del reporte. Estas se muestran bajo un listado con la siguiente información:
        - Nombre del tipo de material

- Consulta de material más buscado por tipo de material: reporte con la información del material más consultado acorde a un tipo de material en específico.
    - Entrada: cadena de caracteres con formato XML con la siguiente información:
        - Llave de seguridad
        - Identificador de sistema consumidor
        - Número de registros por página
        - Página actual
    - Salida: cadena de caracteres con formato XML con el detalle del reporte. Estas se muestran bajo un listado con la siguiente información:
        - Biblioteca

- Título de coincidencia y dirección URL para acceso directo al detalle.
- Autor

3.2.3. Elementos necesarios para desarrollo de sistema

Para el desarrollo del sistema *broker* propuesto, se requiere contar con una serie de componentes indispensables que se detallan a continuación.

3.2.3.1. Conectividad con sistema GLIFOS

Es necesario contar con acceso a los servicios web que provee el software bibliotecario GLIFOS para poder realizar consultas en su base de datos.

3.2.3.2. Sistema de base de datos

Se requiere de un sistema de base de datos relacional con soporte lenguaje SQL para alojar las estadísticas y bitácora de las consultas que se hacen sobre el *broker*. En el caso de que se elija las herramientas que provee Microsoft para desarrollo, se recomienda que tenga soporte para trabajar con Entity Framework de .NET.

3.2.3.3. Integrated development environment (IDE)

Aplicación de software que provee de los medios para el desarrollo de software. Es necesario que cuente con las herramientas para programación de sitios web, servicios web y conexión a bases de datos. Se recomienda el uso de Microsoft Visual Studio por la facilidad de desarrollo y las tecnologías de Windows Communication Foundation y Entity Framework que provee.

3.2.3.4. Servidor web

Se requiere de un servidor web físico o virtual para alojar el sitio cuando este se haga público. Esto se detallará en el capítulo 4 referente al plan de implementación del sistema.

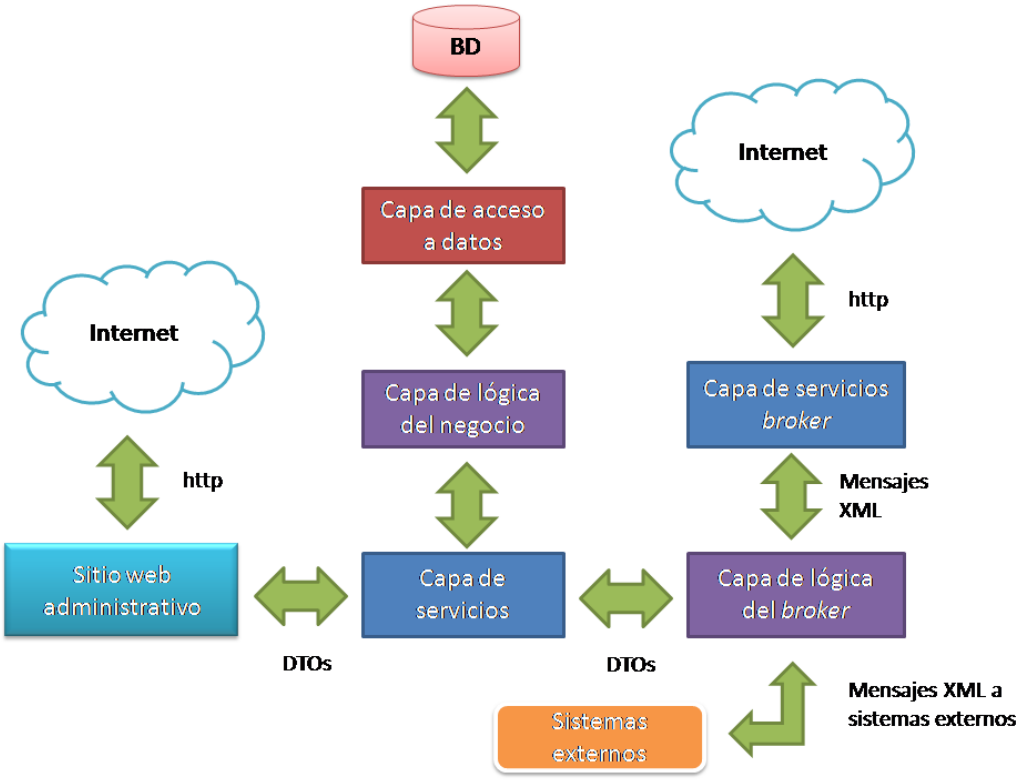
3.3. Diseño

A lo largo del diseño se detalla el diagrama de arquitectura de sistema propuesto así como los diagramas de flujo de datos y de estructuras sugeridos para el mismo.

3.3.1. Arquitectura del sistema

El sistema basará su arquitectura en un modelo de n capas con la finalidad de separar de manera ordenada los distintos componentes que en su totalidad componen el sistema. En la figura 8 se ilustra el diagrama de arquitectura del sistema y en las siguientes secciones se explica cada uno de los distintos componentes del mismo.

Figura 8. **Diagrama de arquitectura de sistema *broker***



Fuente: elaboración propia, con programa de Microsoft Power Point.

3.3.1.1. Capa de acceso a datos

Es la capa encargada de realizar las peticiones directamente a la base de datos. Se encuentra compuesta por un conjunto de clases que internamente contienen las operaciones a realizar sobre la base de datos. Su enfoque es tener una clase por cada tabla en la base de datos. De manera que las operaciones de consulta, inserción, actualización y eliminación en cada tabla puedan hacerse en su clase equivalente. Para los reportes se tendrá una clase independiente.

3.3.1.2. Capa de lógica del negocio

Es la capa encargada de procesar las entradas provenientes de la capa de servicios web del sistema y convertirlas en salidas. Se encuentra compuesta por un conjunto de clases. En esta capa se tiene una clase por tabla en la base de datos. Por consiguiente todas las consultas y operaciones que se solicitan sobre una tabla en la base de datos deben de ser procesadas por su equivalente en esta capa.

Su función es procesar toda la información requerida o a almacenar en la base de datos y retornar el resultado a la capa de servicios. Adicionalmente se encarga de preparar los datos que se generan a partir de las consultas al *broker*, para que puedan ser almacenados en la base de datos. Con ello se tiene una bitácora de lo acontecido en el *broker* a partir del cual se generan los reportes.

3.3.1.3. Capa de servicios

Es el medio de comunicación existente entre la capa de lógica del negocio y las capas web (para el sitio web administrativo) y de lógica del *broker*. Utiliza como lenguaje de comunicación objetos de transferencia de datos (DTOs) los cuales pueden ser interpretados por todas las capas involucradas en la comunicación.

Como es una arquitectura SOA en la que se basa el sistema, la comunicación se realiza a partir de servicios web. Por tal razón, esta capa se compone de un conjunto de servicios web que van acorde a la operación que se desea realizar en la lógica. Por consiguiente se tendrá un servicio web por cada clase de la capa de lógica de negocios.

3.3.1.4. Capa de lógica de *broker*

Capa encargada de transformar y direccionar las consultas que llegan al *broker* y que van destinadas a hacerse en sistemas externos bibliotecarios. Esto quiere decir, que todas las consultas a sistemas externos bibliotecarios entran en un formato XML estándar al *broker*. Luego, la capa transforma la petición (acorde al sistema bibliotecario a consultar) al formato requerido por el sistema bibliotecario externo. De igual manera cuando un sistema bibliotecario externo responde, transforma el mensaje recibido en un mensaje XML que es retornado al solicitante. Toda entrada y salida al *broker* es enviada a partir de objetos de transferencia de datos a la capa de servicios del sistema para que se almacene en la base de datos.

3.3.1.5. Capa de servicios de *broker*

Capa compuesta por un conjunto de servicios web a partir de los cuales se pueden hacer consultas a múltiples sistemas bibliotecarios externos. La comunicación se hace a partir de mensajes XML y cada uno de ellos puede contener petición sobre consultas a uno o más sistemas bibliotecarios. De la misma manera, cada mensaje de salida puede contener información proveniente de uno o más sistemas externos bibliotecarios.

3.3.1.6. Capa web

Es la interfaz visual con la que el usuario final interacciona. Esta capa se compone de un conjunto se páginas web cuya funcionalidad es administrar y presentar al administrador estadísticas sobre las consultas que se realizan al *broker*.

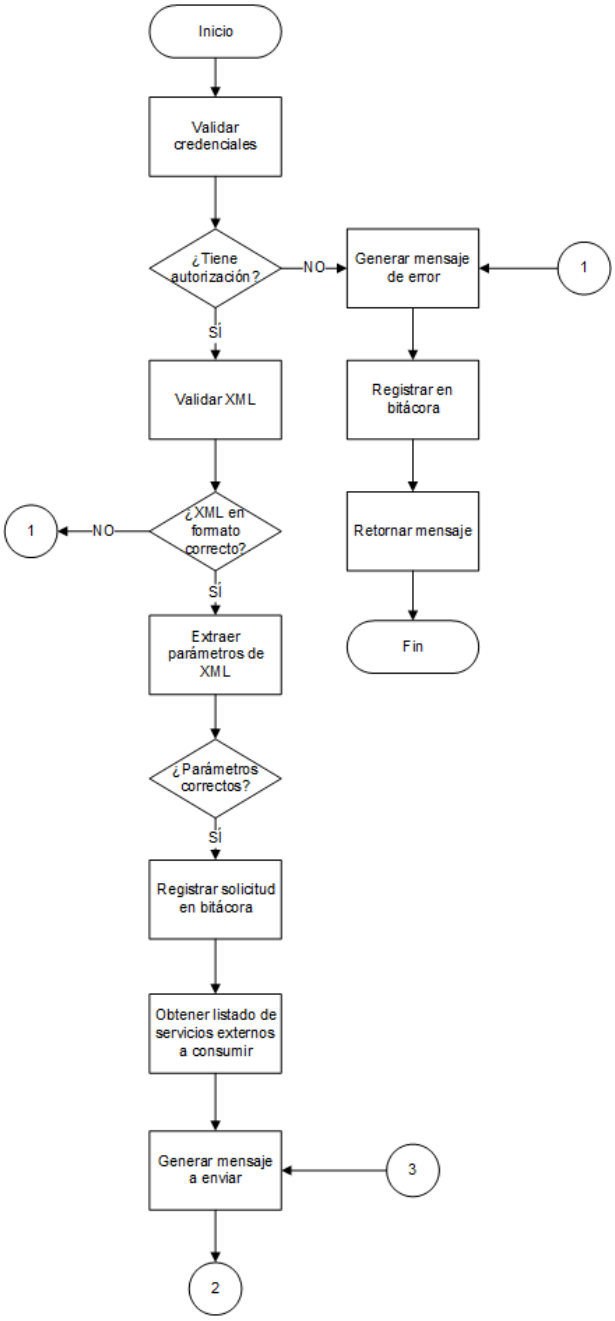
3.3.2. Procesos principales

El sistema cuenta con dos procesos principales que se emplean en todas las operaciones que se llevan a cabo. Estos se detallan a continuación a través de sus respectivos diagramas de flujo (ver figuras 9, 10, 11 y 12).

3.3.2.1. Proceso de consultas a *broker*

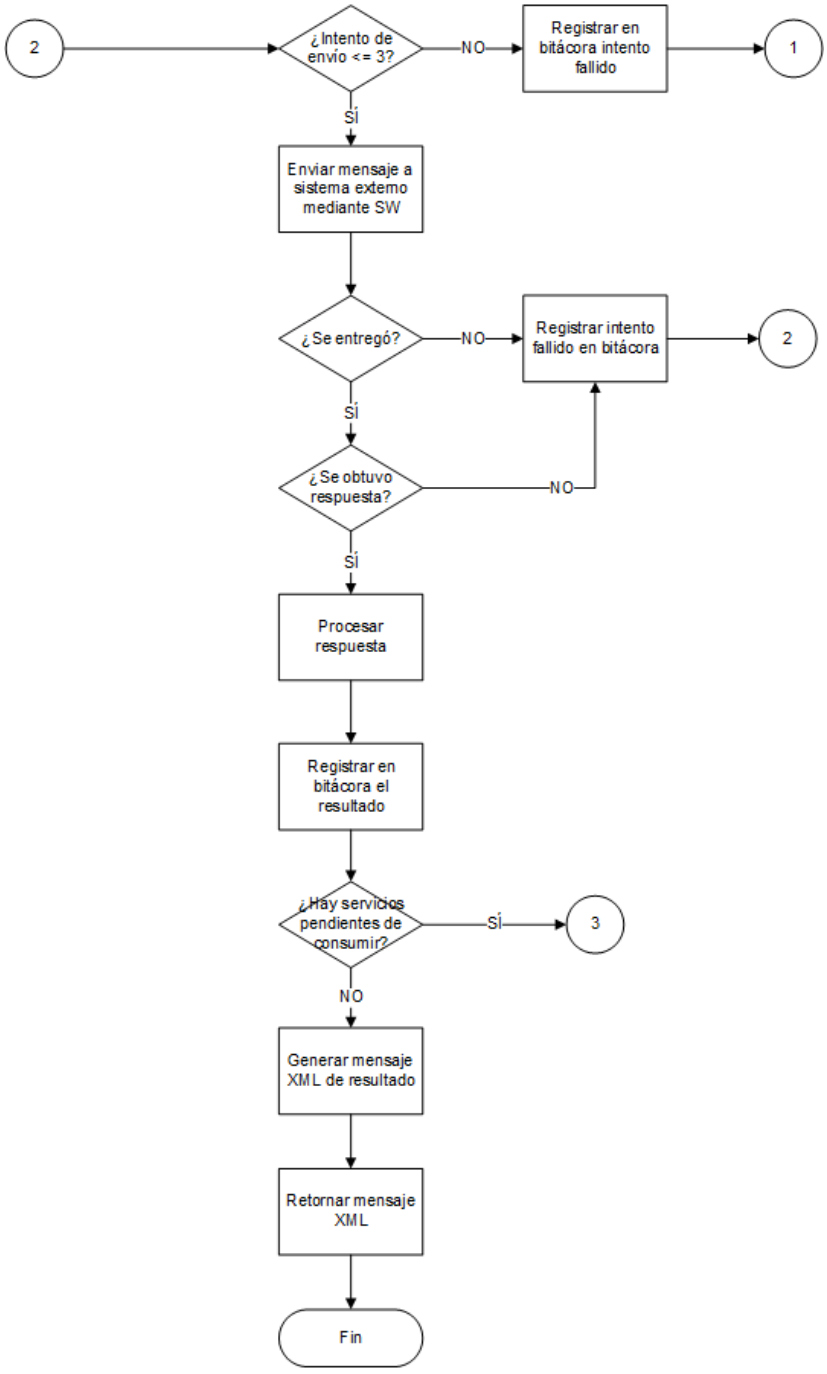
Este proceso abarca todas las consultas que se realizan a sistemas bibliotecarios externos por medio del *broker*. El detalle de los pasos que se realizan a lo largo de este proceso, puede verse en las figuras 9 y 10 que representan un solo diagrama de flujo. Las referencias entre los diagramas de las dos figuras se representan mediante círculos con un número correlativo adentro.

Figura 9. **Diagrama de flujo de consultas a *broker* I**



Fuente: elaboración propia, con programa de Microsoft Visio.

Figura 10. **Diagrama de flujo de consultas a *broker* II**

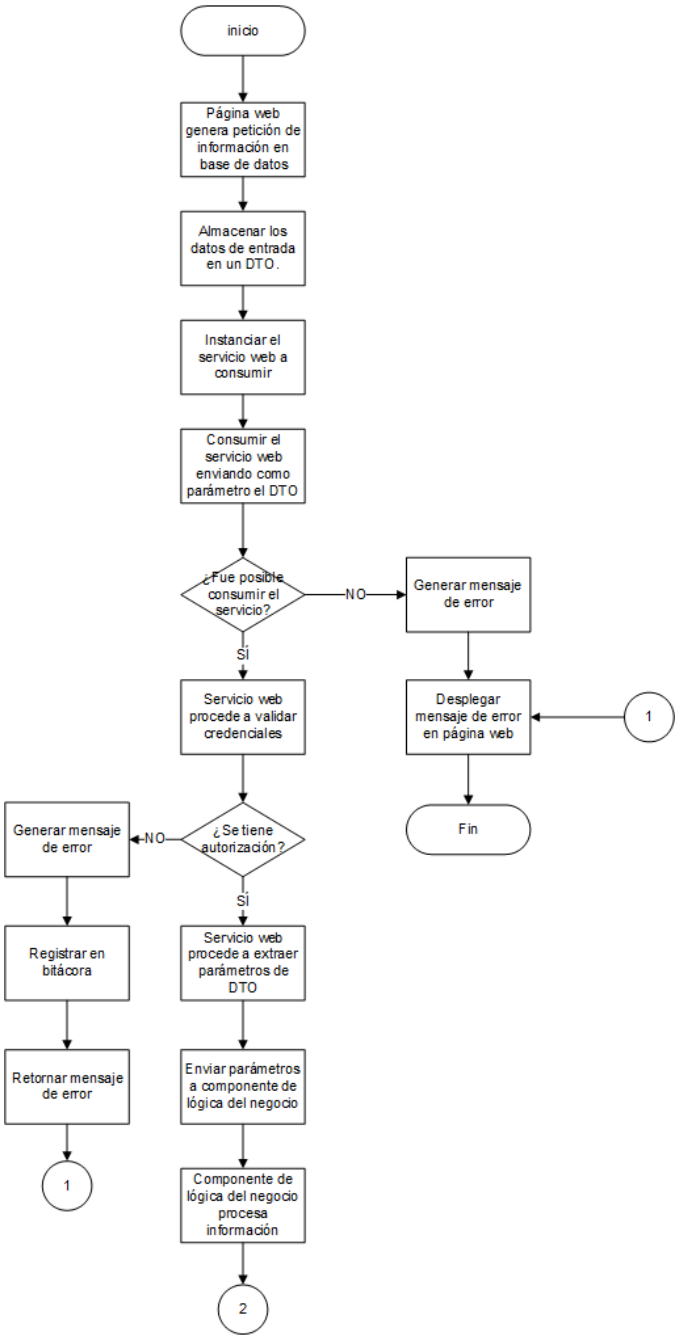


Fuente: elaboración propia, con programa de Microsoft Visio.

3.3.2.2. Proceso de consultas desde sitio web a base de datos

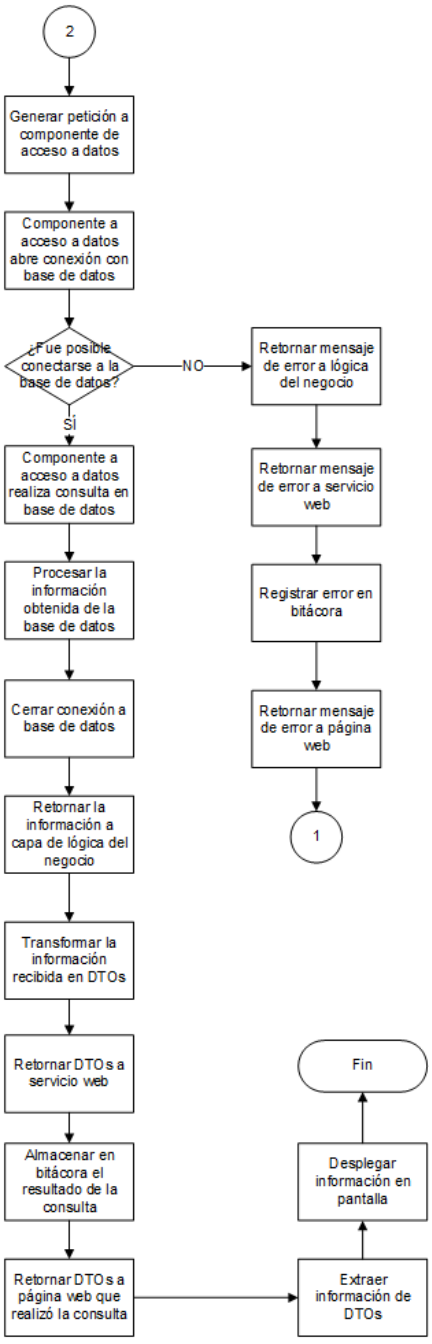
Este proceso abarca todos los requerimientos de información que el sitio web desea hacer a la base de datos. El detalle de los pasos que se realizan a lo largo de este proceso puede verse en las figuras 11 y 12 que representan un solo diagrama de flujo. Las referencias entre los diagramas de las dos figuras se representan mediante círculos con un número correlativo adentro.

Figura 11. **Diagrama de flujo de consultas desde sitio web a base de datos I**



Fuente: elaboración propia, con programa de Microsoft Visio.

Figura 12. **Diagrama de flujo de consultas desde sitio web a base de datos II**

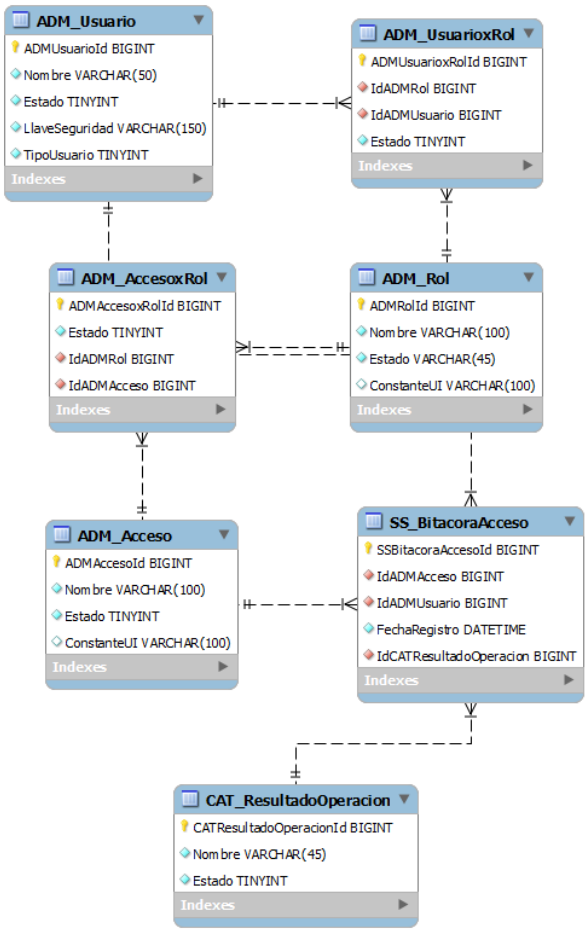


Fuente: elaboración propia, con programa de Microsoft Visio.

3.3.3. Diagrama de estructuras

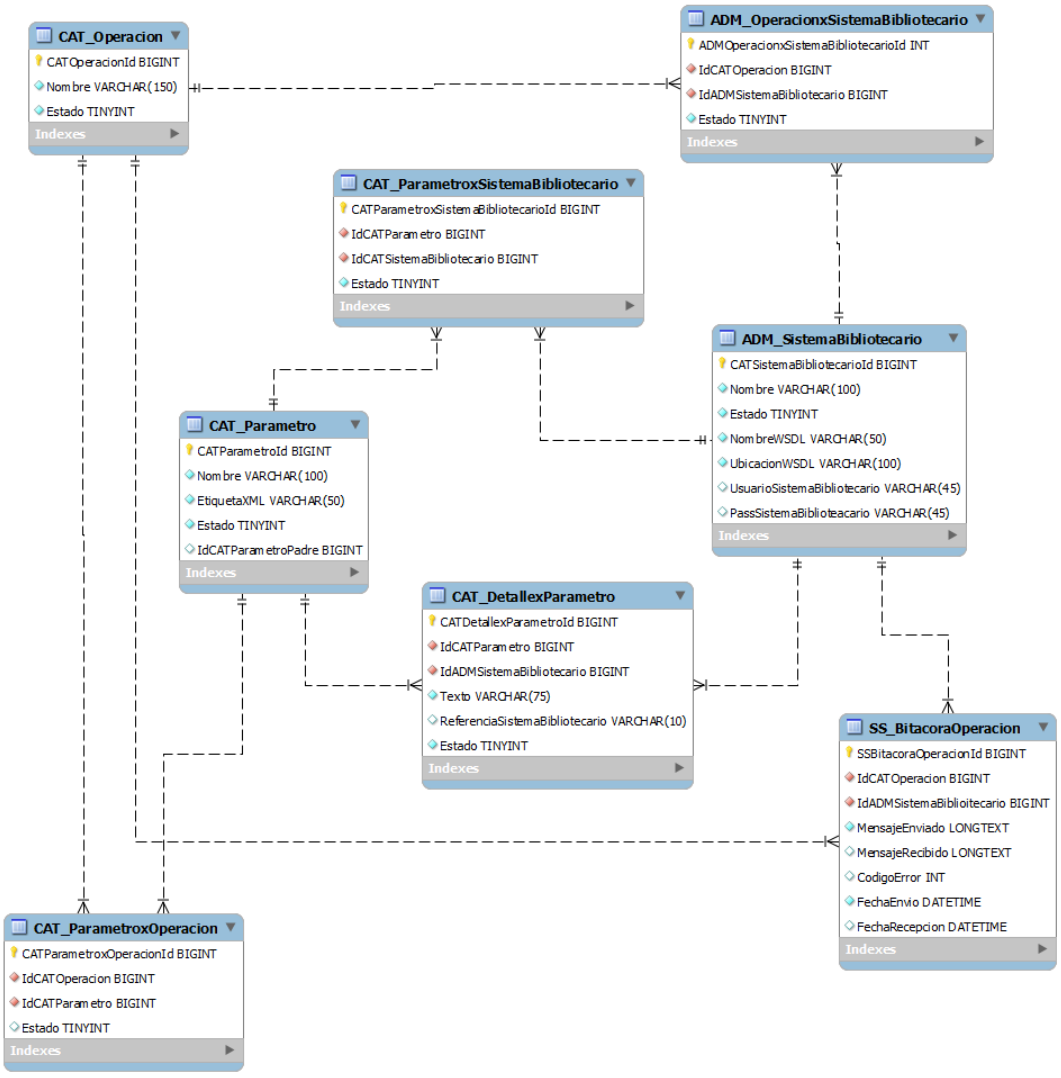
Para el almacenamiento de la información se contará con una base de datos relacional. La información se distribuirá en tablas acorde al diagrama de estructuras que puede verse en las figuras 13 y 14. Ambas figuras representan un solo diagrama.

Figura 13. **Diagrama de estructuras I**



Fuente: elaboración propia, con programa de MySQL Workbench.

Figura 14. **Diagrama de estructuras II**



Fuente: elaboración propia, con programa de MySQL Workbench.

A continuación se detalla cada tabla respecto a su función dentro del modelo entidad relación.

- ADM_Acceso: en esta tabla se almacena el catálogo de accesos para el sitio administrativo. Se compone de los siguientes campos:
    - ADMAccesoId: llave primaria de tipo numérico que identifica un registro en la tabla.
    - Nombre: nombre del acceso
    - Estado: estado del acceso. La tabla funciona a través de una eliminación lógica. Las operaciones de eliminar actualizan el estado del registro a un estado no activo.
    - ConstanteUI: nombre de la constante con la que se identificará este acceso en el sitio web administrativo.

- ADM_AccesoxRol: contiene el detalle de la asignación de accesos a uno o más roles. Se compone de los siguientes campos:
    - ADMAccesoxRolId: llave primaria de tipo numérico que identifica un registro en la tabla.
    - Estado: estado de la asociación de un acceso con un rol. Puede ser activo o no activo (para el caso de la eliminación lógica).
    - IdADMRol: llave primaria de la tabla ADM_Rol que identifica al rol sobre el que se asocia el acceso.
    - IdADMAcceso: llave primaria de la tabla ADM_Acceso que identifica el acceso que se asocia al rol.

- ADM_OperacionxSistemaBibliotecario: contiene el detalle de la asociación de una operación con un sistema bancario. Es decir, el listado de operaciones disponibles para cada sistema bancario. Se compone de los siguientes campos:
    - ADMOperacionxSistemaBibliotecarioId: llave primaria de la tabla que identifica de manera única un registro.

- IdCATOperacion: llave primaria de la tabla CAT_Operacion que identifica la operación que se asocia a un sistema bibliotecario externo.
- IdADMSistemaBibliotecario: llave primaria de la tabla ADM_SistemaBibliotecario que identifica el sistema bibliotecario al que se le asocia una operación.
- Estado: estado en el que se encuentra la relación entre operación y sistema bibliotecario. Esta puede ser activa o no activa (para la eliminación lógica).

- ADM_Rol: catálogo de roles para el sistema. Se compone de los siguientes campos:
  - ADMRolId: llave primaria de la tabla que identifica de manera única un registro.
  - Nombre: nombre del rol
  - Estado: estado en el que se encuentra el rol. Puede ser activo o inactivo (para el caso de la eliminación lógica).
  - ConstanteUI: nombre de constante con la que se identifica al rol en el sitio web.

- ADM_SistemaBibliotecario: catálogo de sistemas bibliotecarios con los que el *broker* tiene comunicación. Está compuesta por los siguientes campos:
  - ADMSistemaBibliotecarioId: llave primaria de la tabla que identifica de manera única un registro.
  - Nombre: nombre del sistema bibliotecario
  - Estado: estado en el que se encuentra el sistema bibliotecario. Puede ser activo o inactivo (para el caso de la eliminación lógica).

- NombreWSDL: nombre del archivo de web service definition language (WSDL) en donde se encuentra la definición del servicio web que se consume para comunicarse con el sistema bibliotecario.
- UbicacionWSDL: directorio en sistema de archivos en donde se encuentra ubicado el archivo WSDL.
- UsuarioSistemaBibliotecario: nombre de usuario con el que el sistema *broker* puede autenticase con el sistema bibliotecario que se registra en esta tabla.
- PassSistemaBibliotecario: contraseña con la que el usuario del sistema *broker* puede autenticarse con el sistema bibliotecario que se registra en esta tabla.

- ADM_Usuario: listado de usuarios del sistema *broker* y del sistema administrativo. Se compone de los siguientes campos:
  - ADMUsuarioId: llave primaria de la tabla que identifica de manera única un registro de la tabla.
  - Nombre: nombre del usuario
  - LlaveSeguridad: contraseña encriptada del usuario
  - TipoUsuario: indica el tipo del usuario. Se identifican los usuarios de tipo administrativo y los usuarios de sistemas externos que desean hacer consultas hacia las distintas bibliotecas.

- ADM_UsuarioxRol: tabla que contiene la relación de los distintos roles asignados a un usuario. Se compone de los siguientes campos:
  - ADMUsuarioxRolId: llave primaria que identifica un registro de manera única en la tabla.
  - IdADMRol: llave primaria de la tabla ADM_Rol que identifica el rol que se asocia al usuario.

- IdADMUsuario: llave primaria de la tabla ADM_Usuario que identifica el usuario al que se asigna el rol.
- Estado: estado de la asociación entre rol y usuario. Puede ser activo o no activo (para la eliminación lógica).

- CAT_DetallexParametro: contiene el catálogo de elementos disponibles para un parámetro. Por ejemplo, si se tuviese un parámetro para tipos de material bibliográfico, dentro de este catálogo se encontrarían las novelas. Se compone de los siguientes campos:
  - CATDetalleParametroId: llave primaria de la tabla que identifica de manera única un elemento del catálogo de parámetros.
  - IdADMSistemaBibliotecario: llave primaria de la tabla ADM_SistemaBibliotecario.
  - Texto: el nombre que lleva el elemento del catálogo de parámetros.
  - ReferenciaSistemaBibliotecario: identificador del elemento del catálogo en el sistema bibliotecario actual.
  - Estado: estado de los elementos de la tabla. Puede ser activo o no activo (para el caso de la eliminación lógica).

- CAT_Operacion: catálogo de las distintas operaciones que maneja el sistema. Se compone de los siguientes campos:
  - CAT_OperacionId: llave primaria con la que se identifica de manera única un registro en la tabla. perfil.
  - Nombre: nombre de la operación
  - Estado: estado del elemento de catálogo. Puede ser activo o no activo (para la eliminación lógica).

- CAT_Parametro: catálogo de parámetros sobre los que se puede hacer consulta a sistemas externos. Por ejemplo, un elemento del catálogo podría ser el nombre del autor. Sobre este parámetro pueden realizarse búsquedas también. Se compone de los siguientes campos:
    - CATParametroId: llave primaria que identifica de manera única un registro en la tabla.
    - Nombre: nombre del parámetro
    - EtiquetaXML: nombre de la etiqueta XML que lo identifica en los campos de entrada y salida hacia y desde el *broker*.
    - Estado: estado del parámetro que puede ser activo o no activo (para el caso de la eliminación lógica).
    - IdCATParametroPadre: para el caso que se tenga un parámetro (etiqueta XML) que se encuentre dentro de otra etiqueta XML, la segunda etiqueta funciona como el padre de la actual.

- CAT_ParametroxOperacion: catálogo de parámetros para cada operación dentro del sistema. Se compone de los siguientes campos:
    - CATParametroxOperacionId: llave primaria de la tabla que identifica un registro de manera única.
    - idCATOperacion: llave primaria de la tabla CAT_Operacion que identifica de manera única una operación en el sistema.
    - IdCATParametro: llave primaria de la tabla CATParametro
    - Estado: estado en el que se encuentra la relación entre un parámetro y una operación Puede ser activo o no activo.

- CAT_ParametroxSistemaBibliotecario: contiene el listado de asociaciones entre sistemas bibliotecarios y parámetros. En este caso determina qué parámetros se utilizan en el sistema bibliotecario. Contiene las siguientes columnas:

- CATParametroxSistemaBibliotecarioId: llave primaria que identifica de manera única un registro en la tabla.
- IdCATParametro: parámetro que se asocia a un sistema bibliotecario en particular.
- IdCATSistemaBibliotecario: sistema bibliotecario al que se desea asignar los parámetros del sistema.
- Estado: estado de la relación entre parámetro y sistema bibliotecario. Puede ser activo o no activo (para el caso de la eliminación lógica).

- CAT_ResultadoOperacion: catálogo de los posibles resultados de operación. Se compone de los siguientes campos:
  - CATResultadoOperacionId: llave primaria del resultado de operación.
  - Nombre: nombre del resultado de la operación
  - Estado: estado en el que se encuentran los registros de la tabla CAT_ResultadoOperacion. Pueden ser activos o no activos (para el caso de la eliminación lógica).

- SS_BitacoraAcceso: registra las operaciones de consulta de accesos en la base de datos. Se compone de las siguientes columnas:
  - SSBitacoraAccesoId: llave primaria de la tabla que identifica de manera única un registro.
  - IdADMAcceso: llave primaria del acceso sobre el cual se registra la bitácora.
  - IdADMUsuario: llave primaria del usuario que realiza la operación de consulta de acceso.
  - FechaRegistro: fecha y hora en la que se realiza la consulta.

- IdCATResultadoOperacion: llave primaria que especifica cuál fue el resultado de la operación realizada.

- SS_BitacoraOperacion: registra las consultas que se hacen al *broker*. Se compone de los siguientes campos:
  - SSBitacoraOperacionId: llave primaria de la tabla que identifica un registro de manera única.
  - IdCATOperacion: llave primaria de la operación que se realizó en el *broker*.
  - IdADMSistemaBibliotecario: llave primaria del sistema bibliotecario al que se consultó en el *broker*.
  - MensajeEnviado: cadena de caracteres con el mensaje XML que se envió al sistema bibliotecario externo sobre el que se realizó la consulta.
  - MensajeRecibido: cadena de caracteres con el mensaje XML que se recibió de parte de un sistema bibliotecario externo.
  - CodigoError: código de error retornado por el servicio web del sistema bibliotecario externo.
  - FechaEnvio: fecha y hora en la que se envió el mensaje al sistema bibliotecario externo.
  - FechaRecepcion: fecha y hora en la que se obtuvo repuesta de parte del sistema bibliotecario externo al que se hizo la consulta.

4. IMPLEMENTACIÓN SUGERIDA

En el presente capítulo se presenta el conjunto de tareas necesarias para la instalación y pruebas del sistema *broker*. Este fue diseñado bajo una arquitectura SOA para intercomunicar múltiples sistemas bibliotecarios. En este caso se hace referencia a la interconexión del sistema con el software GLIFOS. Sin embargo, es importante resaltar que también queda abierta la disponibilidad de conexión con otros tipos de software bibliotecario que utilicen arquitecturas similares.

Para lo que es la implementación de sistema se provee de una lista detallada de eventos o actividades requeridas, de tal manera que se entrega un mapa detallado de las fases de implementación a cualquier organización, persona individual o cualquier interesado que desee ponerlo en práctica.

4.1. Alcance

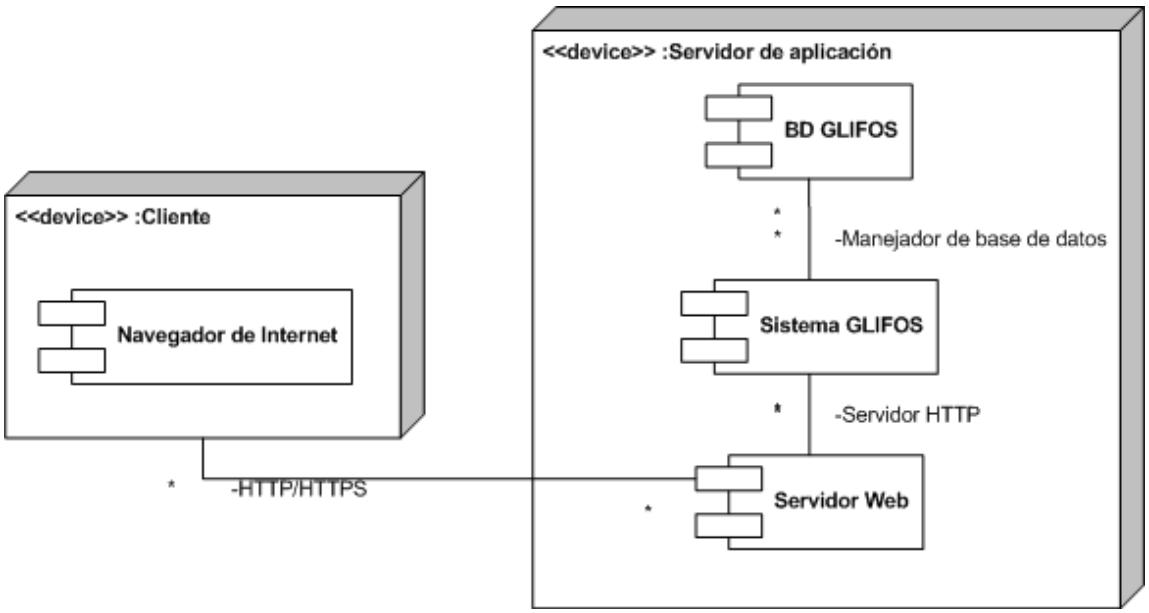
El plan de implementación, se inicia realizando una breve descripción de la interacción actual que tiene el sistema GLIFOS para luego proponer la interacción con el sistema propuesto. De la misma manera se listan las diferentes fases de la implementación dentro de la arquitectura física del lugar en el que se hará la instalación.

Luego se describen las diferentes pruebas en las configuraciones y desenvolvimiento del sistema en un ambiente de producción. Se finaliza con el entrenamiento y capacitación a las distintas personas involucradas.

4.2. Estudio de las condiciones actuales

Como se menciona en el capítulo 2, el sistema actual está integrado por un servidor de aplicaciones en el cual se encuentra instalado el sistema GLIFOS, su base de datos y un servidor web. Como cliente puede tenerse cualquier dispositivo que cuente con un navegador de internet, acceso a la red y la comunicación se realiza por medio del protocolo HTTP o HTTPS. En la figura 15 se presenta el diagrama de despliegue del sistema GLIFOS.

Figura 15. **Diagrama de despliegue del sistema GLIFOS**

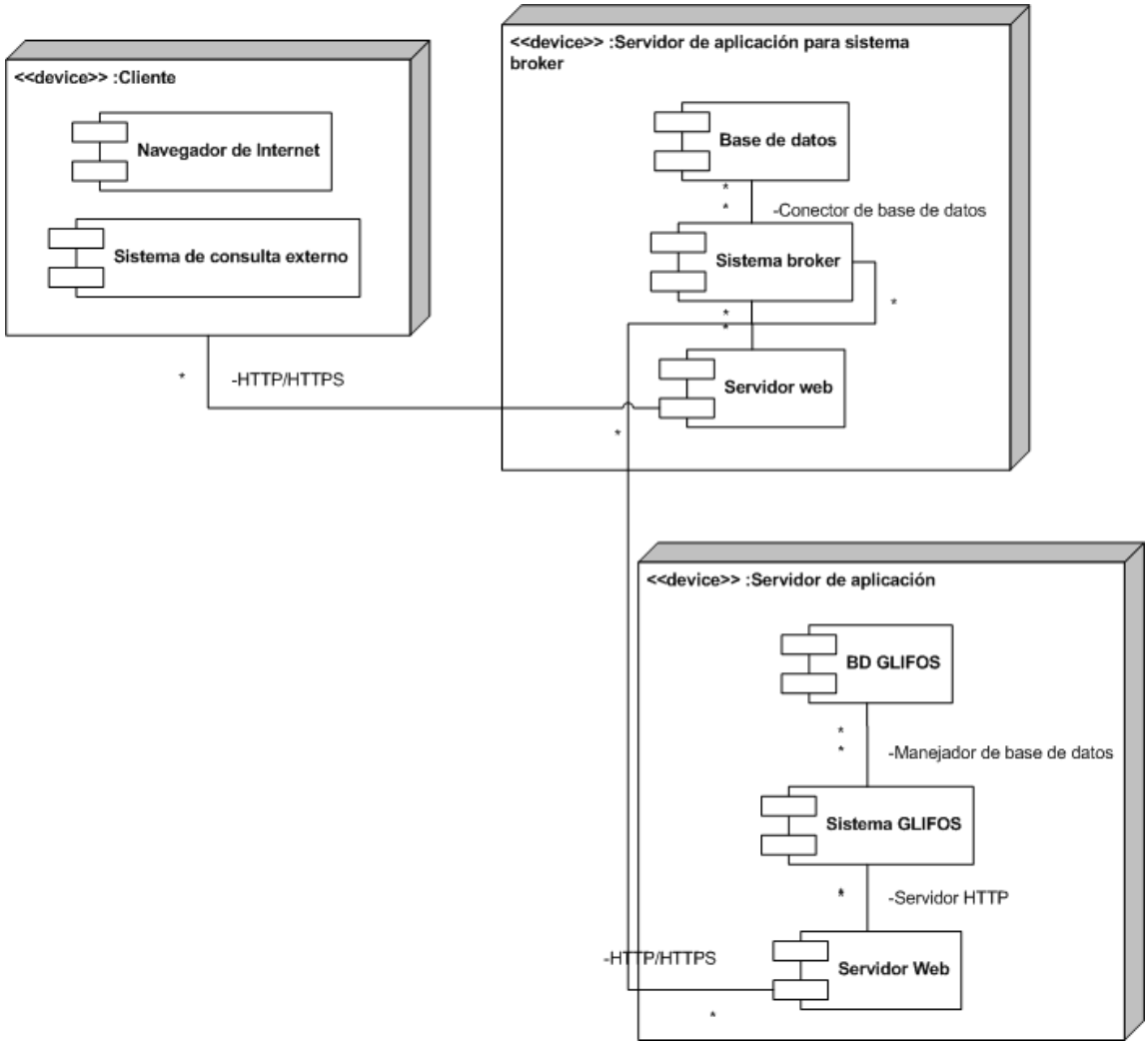


Fuente: elaboración propia, con programa de Microsoft Visio.

Este despliegue básico es el que comúnmente se encuentra instalado en bibliotecas que utilizan el sistema GLIFOS. Por ejemplo se tiene el caso de la Biblioteca Central de la Universidad de San Carlos de Guatemala. En la figura 16 se muestra como el sistema propuesto tiene relación con el despliegue

original. Es decir, se incorpora al diagrama de despliegue el sistema *broker* y como éste interactúa con GLIFOS.

Figura 16. **Diagrama de despliegue del sistema *broker***



Fuente: elaboración propia, con programa de Microsoft Visio.

En este diagrama el cliente puede comunicarse al sistema *broker* a través de dos formas:

- Mediante un navegador de internet, el cual hace peticiones a través del protocolo HTTP/HTTPS con relación al sitio administrativo. De modo que pueda obtener estadísticas u otra información en base a la utilización del *broker*.

- Mediante un sistema de aplicación que consume servicios web del *broker*, para hacer consultas en distintos sistemas bibliotecarios. Esto es a través de la capa de servicios del sistema *broker*.

4.3. Planificación de las etapas de implementación

Dentro de la planificación de la implementación, se abarca como primer punto la asignación de responsabilidades dentro de las personas involucradas en el despliegue. Luego se procede a detallar un cronograma con el listado de actividades involucradas en la instalación.

Finalmente se abarcan los temas relacionados a los recursos o material a entregarse luego del proceso de implementación del sistema así como los requerimientos de hardware recomendados para el equipo de cómputo que albergará al sistema.

4.3.1. Responsabilidades

Es necesario establecer un compromiso con las personas que implementarán el sistema. Para este fin se propone designar a una persona como administradora del proceso de implementación. Este tendrá dentro de sus responsabilidades: conocer la arquitectura física sobre la que se instalará la aplicación, administrar el tiempo y otros recursos en el momento de la

implementación. También estará a cargo de reportar errores y desafíos en el proceso de implementación.

El administrador puede apoyarse en un equipo de trabajo para facilitar y agilizar el proceso. Esto a través de la delegación de responsabilidades, tomando en cuenta que él será el encargado principal y responsable del éxito o fracaso de este proceso. El tamaño del equipo queda a consideración de administrador, pero se recomienda un equipo no mayor a 3 personas. Esto debido a que el sistema a implementarse no es un sistema demasiado grande y/o complejo. Si se tiene a muchas personas involucradas, puede que llegue a desperdiciarse cierto parte del recurso asignado.

4.3.2. Cronograma

A continuación se propone el cronograma aproximado con las tareas a realizar durante la implementación del sistema *broker*.

- Adecuación del lugar físico donde se alojará el equipo de cómputo físico. Esto incluye red eléctrica, cableado estructurado de red, establecer las condiciones de temperatura y seguridad en los equipos. Se estiman 5 días hábiles para la adecuación del lugar.

- Visualización de los recursos necesarios de software y hardware, para que el sistema pueda implantarse de manera correcta. Para realizar esta tarea se estima 1 día hábil.

- Instalación y configuración de herramientas necesarias en los equipos físicos que albergarán la aplicación. Entre estas se tiene el sistema

operativo, servidor web, sistema administrador de base de datos. Para completar con esta tarea se estima un tiempo de 2 días hábiles.

- Instalación de *scripts* de base de datos y realización de pruebas para verificación de que estén completos los catálogos y toda la información necesaria, para que pueda ejecutarse el sistema *broker*. Se estima 1 día para completar esta tarea.

- Instalación de sistema. En esta parte se copian los archivos que componen el sistema al servidor físico. Se realizan los cambios necesarios de ambiente de pruebas a ambiente de producción de modo que las configuraciones del sistema queden establecidas de manera fija. Se estima 1 día para completar esta tarea.

- Pruebas de conexiones entre *broker* y sistemas bibliotecarios externos. Se estima un tiempo de 5 días para verificar que todas las conexiones con sistemas bibliotecarios externos se realicen de manera adecuada.

- Pruebas de consumo de los servicios de *broker*. Estas pruebas simulan sistemas externos, consumiendo los servicios que provee el *broker*. Se estima un tiempo de 3 días para completar esta tarea.

- Pruebas del sitio administrativo. Se prueba que toda la funcionalidad de sitio sea funcional. Se estima un tiempo de 2 días para completar esta tarea.

- Luego de tener el sistema totalmente operativo, se realizará el proceso de entrenamiento y capacitación a los usuarios reales del sistema. Se tiene un tiempo estimado de 5 días para llevar a cabo la tarea.

- Finalmente se tiene el soporte en el cual se asistirá a los usuarios en el uso correcto del sistema así como en posibles eventualidades que puedan ocurrir. Especialmente en el caso que se detecten fallas en el tiempo de vida del software. El tiempo de soporte queda a criterio del administrador de la implementación.

4.3.3. Recursos

Entre los recursos que debe de entregarse al momento de la implementación se encuentra el manual técnico del sistema. En este se encuentran detallados los elementos de arquitectura, configuraciones a realizar en archivos de instalación, interacción del *broker* con sistemas externo (detalle de los servicios web) y documentación de código fuente.

Adicionalmente se debe de entregar un manual de implementación compuesto por el manual de instalación y el cronograma de actividades.

Por último se proveerá de un manual de usuario en el cuál se encontrarán descritas las funcionalidades del sistema, así como las formas correctas de uso y descripción de mensajes de error.

4.3.4. Hardware

Los requisitos de hardware necesarios para la implementación de sistema desarrollado deben ser de una tecnología relativamente reciente. Sin embargo, este puede no ser de tecnología de última generación. Esto debido a que los niveles de estrés a los que se someterá el sistema, no serán demasiado altos. Se recomiendan las siguientes características de hardware en el equipo en donde se hará la instalación:

- Procesador con múltiples núcleos de al menos 2 GHz. y 4 MB de memoria caché.

- 2 GB de memoria RAM

- Disco duro con espacio disponible mínimo de 50 GB.

- Adaptador de red de 1 Gbps y 2 puertos

4.4. Pruebas y correcciones menores

Como parte del proceso de implementación se incluye una etapa para realización de pruebas y correcciones menores en problemas que se detecten durante esta fase. Los escenarios de prueba se centran en tres tipos de eventos:

- Conexiones entre *broker* y sistemas bibliotecarios externos

- Consumo de servicios web del *broker*

- Pruebas en sitio web administrativo.

4.4.1. Pruebas de conexiones entre *broker* y sistemas bibliotecarios externos

Se procederá a realizar la configuración necesaria para que el sistema pueda tener comunicación con todos los sistemas bibliotecarios externos. Se recomienda: la elaboración de un plan de pruebas en el que se incluya todas las

interfaces disponibles, así como una colección de datos de prueba sobre los que se harán las consultas a sistemas externos. Las operaciones a probar para los sistemas externos son las siguientes:

- Consulta básica a catálogo
- Consulta avanzada (búsqueda booleana)
- Consulta de estado en biblioteca
- Consulta de material por autor
- Consulta de material por tema
- Consulta para ver detalle de coincidencia

4.4.2. Pruebas de consumo de los servicios de *broker*

Se recomienda elaborar un plan de pruebas que incluya el identificar escenarios puntuales para las operaciones que el *broker* provee a sistemas de consulta. Cada uno de estos escenarios debe de contar con sus propios datos de prueba para verificar su correcto funcionamiento. Las operaciones disponibles son las siguientes:

- Consulta básica a catálogo
- Consulta avanzada (búsqueda booleana)
- Consulta de estado en biblioteca

- Consulta de material por autor
- Consulta de material por tema
- Consulta para ver detalle de coincidencia

4.4.3. Pruebas del sitio administrativo

El último punto relacionado a las pruebas del sistema se centra en verificar el funcionamiento adecuado del sitio administrativo. Para esto se recomienda la elaboración de un plan de prueba en el que se incluya los siguientes escenarios:

- Inicio de sesión al sistema
- Validación de credenciales
- Carga de accesos dentro del sistema
- Creación de usuarios
- Asignación de acceso a usuarios
- Cambio de contraseña
- Consulta de libros más buscados
- Consulta de autores más buscados

- Consulta de temas más buscados

- Consulta de tipo de material más consultado

- Consulta de material más buscado por tipo de material

Para cada uno de estos escenarios se necesita contar con datos de prueba y un listado de resultados esperados para el caso de los reportes.

4.5. Entrenamiento

El entrenamiento no debe de ser muy largo ni costoso, ya que la interfaz de usuario no es compleja. Esto debido a que el principal enfoque del sistema se encuentra en la interacción entre sistemas bibliotecarios. El entrenamiento abarca únicamente el módulo administrativo, para el cual se debe de realizar una sesión informativa para explicar el sistema a grandes rasgos.

Luego de esta sesión general y según el perfil de cada uno de los usuarios, se realizarán sesiones personalizadas de modo que el entendimiento del sistema sea mayor. También incluye la entrega de manuales de usuario así como una inducción de los mismos para resolver cualquier duda que exista en su comprensión.

Dicha capacitación debe de ser presencial y utilizando la herramienta de modo que el proceso de aprendizaje sea efectivo y garantizando la familiarización de los usuarios con el sistema.

CONCLUSIONES

1. Los conceptos que provee el patrón de diseño de arquitectura orientada a servicios (SOA) pueden aplicarse a una gran variedad de proyectos de software, no limitándose únicamente al diseño de *brokers*. Toda aplicación web puede ser construida bajo una arquitectura SOA.

2. GLIFOS por ser un software aún en crecimiento no cuenta con tanto material de apoyo en internet. Las únicas fuentes disponibles de información se obtuvieron del sitio web de InfoLib (empresa desarrolladora). Se tuvo un acercamiento con la Biblioteca de la Universidad de San Carlos de Guatemala pero derivado a los derechos de autor de InfoLib, no se pudo obtener ningún tipo de información por ese medio.

3. Hoy en día el software GLIFOS ya cuenta con la funcionalidad de integrar a su catálogo información de otras bibliotecas mediante el proceso *copy-cataloging* del módulo Capture X. Sin embargo esta integración es fuera de línea, es decir, los catálogos de copian a la base de datos de GLIFOS y toda consulta se hace sobre esta base de datos. Por consiguiente, la funcionalidad del *broker* es de gran utilidad, puesto que las consultas se harían directamente en las bases de datos de las bibliotecas consultadas.

4. Considerando que por el momento son muy pocas las bibliotecas en la Universidad de San Carlos que cuentan con un catálogo en línea, se tiene una gran oportunidad para que en conjunto con otras facultades

se realice una propuesta formal que podría llegar a implementarse a un mediano plazo. Esto contribuirá en tener un sistema unificado para consultas en las distintas bibliotecas de la Universidad.

5. El diseño del sistema *broker* se acopló a los conceptos definidos por SOA tanto para el sitio administrativo de *broker* como para el mismo *broker*. Es posible concluir que toda la comunicación hacia y desde la lógica interna del sistema se hace a través de servicios SOA.

6. El patrón de diseño de *broker forwarding* funciona perfectamente para el diseño propuesto ya que las llamadas a sistemas bibliotecarios externos que se realizan al *broker* son redireccionadas de manera transparente a los sistemas destinatarios. De la misma manera las respuestas obtenidas también son enviadas a los solicitantes de manera transparente. Bajo este punto se cumple el patrón de diseño puesto que el *broker* funciona como un intermediario y únicamente encargado de enviar las consultas a quién corresponda.

7. En base al objetivo de aplicar el diseño de arquitectura en capas, este pudo lograrse a través de la segmentación de las tareas del sistema en diversos componentes. Esta es una práctica que debería de implementarse puesto que ayuda a la reutilización de componentes (por capa), así como a facilitar el mantenimiento de los mismos.

8. Es de gran ayuda el empleo de diagramas, para representar el sistema a implementar y con ello mejorar la comprensión del mismo. En este caso se modeló la arquitectura, los procesos principales (para conocer el paso a paso de los mismos), el modelo de estructuras a utilizar para almacenar la información recopilada del *broker* y el diagrama de

implementación que representa la ubicación física en donde se ubicará el sistema.

9. Los conceptos de SOA pueden aplicarse en cualquier lenguaje de programación web que soporte la implementación de servicios web.

10. Contar con un buen plan de implementación que detalle la planificación de las etapas del plan, las responsabilidades de los distintos actores y el cronograma de actividades es de gran ayuda para una adecuada instalación de un software en un ambiente de producción. Esto debido a que se dimensionan las tareas a realizar, el tiempo necesario para completar la instalación y los recursos necesarios.

11. La implementación de un *broker* permite la comunicación de sistemas informáticos heterogéneos de manera transparente, es decir, los sistemas no necesariamente deben de conocer la ubicación de los sistemas con los que se desean comunicar, únicamente hacen la petición al *broker* y es este quien la remite al sistema destinatario. En el caso en que un sistema cambie su ubicación, únicamente el *broker* sufriría cambios y no los demás sistemas que se comunican con este.

RECOMENDACIONES

1. Fomentar el estudio de los conceptos establecidos por SOA, así como los diferentes patrones de diseño que provee para su posible aplicación en otras áreas.

2. Fomentar la creación de proyectos informáticos en pro de la Universidad de San Carlos de Guatemala, ya que al día de hoy muchas de las tareas administrativas se llevan a cabo con documentos físicos.

3. Apoyar el diseño de sistemas informáticos, con la mayor cantidad de diagramas posibles para facilitar la comprensión de los mismos.

4. Investigar cómo se aplican los conceptos de SOA a tecnologías de desarrollo recientes como lo son Android, Blackberry, entre otros.

5. Tomar la iniciativa del sistema propuesto para motivar a las distintas facultades de la Universidad de San Carlos de Guatemala a invertir en tecnologías de la información, que serán de beneficio para todos los estudiantes y que a su vez fomentarán la creación de nuevas propuestas de tecnología entre los estudiantes.

BIBLIOGRAFÍA

1. ANON. *GLIFOS library 7.0.* INFOLIB, S.A, 2010. 12 p.

2. BISKE, Todd. *SOA governance; the key to successful SOA adoption in your organization*. Reino Unido: Packt Publishing, 2008. 214 p.

3. CHOU, David; DEVADOSS, John; ERL, Thomas; et al. *SOA with .NET and Windows Azure: realizing service-orientation with the Microsoft platform.* Estados Unidos: Prentice-Hall; SOA Systems Inc., 2010. 871 p.

4. CLEMENTS, Paul; BACHMANN, Felix; BASS, Len; et al. *Documenting software architecture; views and beyond.* 2a ed. Estados Unidos: Addison-Wesley, 2011. 537 p.

5. ERL, Thomas. *El manifiesto SOA comentado; comentarios y visión sobre el manifiesto SOA.* Traducido del inglés por GUARÍN V., Iván Alfonso; CHAIX, Yves. [ref. noviembre de 2009]. Disponible en Web: <http://www.soa-manifesto.com/annotated_spanish.html>.

6. GOMAA, Hassan. *Software modeling & design; UML, use cases, patterns & software architectures.* Estados Unidos: Cambridge University Press, 2011. 550 p.

7. HOLLEY, Kerry, ARSANJANI, Dr. Ali. *100 SOA questions: asked and answered.* Estados Unidos: Prentice-Hall, 2011. 242 p.

8. SCHMUTZ, Guido; LIEBHART, Daniel; WELKENBACH, Peter. *Service oriented architecture: an integration blueprint; a real world SOA strategy for the integration of heterogeneous enterprise systems.* Reino Unido: Packt Publishing, 2010. 221 p.

9. STAL, Michael. *Using architectural patterns and blueprints for service-oriented architecture*. Vol. 23 No. 2. Estados Unidos: IEEE Computer Society, 2006. [ref. 21 de agosto de 2011]. Disponible en Web: <http://www.stal.de/downloads/ieeesoa.pdf>.

10. THURAISINGHAM, Bhavani. S*ecure semantic service-oriented systems.* Estados Unidos: Taylor and Francis Group, LLC, 2011. 437 p.