UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

F A C U L T A D DE I N G E N I E R Í A

SISTEMA DE PAGOS MÓVILES

Informe de trabajo profesional

Para obtener el título de

INGENIERO EN COMPUTACIÓN

Presenta:

GUILLERMO CONTRERAS PRADO

Director:

A. ADOLFO MILLÁN NÁJERA

Ciudad Universitaria **Marzo 2016**

Agradecimientos

Antes que nada, quisiera agradecer a mis padres y a mis dos hermanas por todo el apoyo, paciencia y cariño durante todos estos años. Ellos han sido mi cimiento y mi motivación para todo lo que he logrado.

También agradecer a todos mis amigos y personas especiales que conocí durante mis años universitarios. Todos ellos me dieron ese impulso extra que necesitaba para poder terminar este proceso.

Agradecer a Gemalto por permitirme realizar este informe y de darme la oportunidad de trabajar con ellos y de tener un gran crecimiento profesional a pesar de no tener un título como aval.

Finalmente, a todos mis profesores y asesores con los que trabajé en la universidad. A pesar de que no siempre hubo una relación estrecha con ellos, les agradezco sinceramente por hacerme crecer como persona.

Contenido

Introducción 1

Objetivo 1

Telefonía celular 1

Usuarios de telefonía celular en México 3

Teléfonos Smartphone 4

Aplicaciones móviles 4

Uso de tecnologías en sistemas bancarios 6

Instituciones de regulación bancaria 8

Contexto profesional 10

Capítulo 1: Organigrama 12

Capítulo 2: Descripción de proyectos 14

Capítulo 3: Proyecto principal (Implementación genérica de Sistema de pagos móviles) 21

Instalación 21

Pruebas de conectividad 40

Pruebas DEV2DEV 43

Pruebas integrales 46

Pruebas UAT 48

Pruebas de estrés y alta disponibilidad 50

Cambios de requerimiento 52

Pase a producción 52

Capítulo 4: Resultados y aportaciones 53

Conclusiones 55

Glosario 56

Referencias 57

Anexos 60

Mensajes ISO 8583 60

Creación de cola JMS en Weblogic (Consola de administración) 63

Características de JDK 6 y JDK 7 64

Oracle RAC 65

SOAP UI 66

Introducción

Objetivo

Colaborar en las tareas de implementación, verificación, validación, administración y actualización del sistema de pagos móviles en las diferentes etapas y ambientes de pruebas definidos, con el propósito de obtener, detectar y resolver las problemáticas que surjan y estabilizar el sistema con el fin de recibir las autorizaciones necesarias por parte del cliente para su paso al ambiente productivo y que el producto sea utilizado por el público en general.

Telefonía celular

Las tecnologías inalámbricas están teniendo mucho auge y desarrollo en estos últimos años, una de las que ha tenido un gran desarrollo ha sido la telefonía celular, desde sus inicios a finales de los setentas ha revolucionado enormemente las actividades que realizamos diariamente. Los teléfonos celulares se han convertido en una herramienta primordial para la gente común y de negocios, las hace sentir más seguras y las hace más productivas. A pesar que la telefonía celular fue concebida para la voz únicamente, debido a las limitaciones tecnológicas de esa época, la tecnología celular de hoy en día es capaz de brindar otro tipo de servicios tales como datos, audio y video con algunas limitaciones, pero la telefonía inalámbrica del mañana hará posible aplicaciones que requieran un mayor consumo de ancho banda.

El servicio de mensajes cortos o SMS (*Short Message Service*) es un mecanismo de entrega de mensajes cortos sobre redes móviles. El mensaje (sólo texto) del dispositivo origen es almacenado en una central SMS para después ser direccionado al dispositivo destino. Esto significa que cuando el receptor no está disponible, el mensaje es guardado para ser enviado después. Cada SMS no puede ser mayor a 60 caracteres. Estos caracteres pueden ser texto (alfanumérico) o mensajes binarios no-texto. Una característica de los SMS es que regresan recibos; esto significa que el origen, si lo desea, puede obtener pequeños mensajes notificando si el SMS fue entregado a su destino.

Dado que los SMS usan canales señalizados en vez de canales dedicados, estos mensajes pueden ser recibidos/enviados simultáneamente con el servicio de voz y datos sobre una red GSM (*Global System for Mobile*). SMS soporta roaming nacional e internacional; por consiguiente se puede enviar un mensaje SMS a cualquier otro teléfono GSM alrededor del mundo. Siendo que tanto GSM, CDMA (*Code Division Multiple Access*) y TDMA (*Time Division Multiple Access*) soportan la tecnología SMS, se puede considerar un servicio universal de datos móviles.

La red GSM también se denomina estándar "de segunda generación" (2G) porque, a diferencia de la primera generación de teléfonos portátiles, las comunicaciones se producen de un modo completamente digital. El estándar GSM permite un rendimiento máximo de 9,6 kbps, que permite transmisiones de voz y de datos digitales de volumen bajo.

La red 3G es tipificada por la convergencia de la voz y datos con acceso inalámbrico a Internet, aplicaciones multimedia y altas transmisiones de datos. Los protocolos empleados en los sistemas 3G soportan más altas velocidades de información enfocados para aplicaciones más allá de la voz tales como audio (MP3), video en movimiento, video conferencia y acceso rápido a Internet, sólo por nombrar algunos.

3G es la abreviación de tercera generación de transmisión de voz y datos a través de telefonía móvil mediante UMTS (*Sistema universal de telecomunicaciones móviles)*. La estructura de redes UMTS está compuesta por dos grandes subredes: la red de telecomunicaciones y la red de gestión; la primera es la encargada de sustentar el transporte de información entre los extremos de una conexión; la segunda tiene como misiones la provisión de medios para la facturación y tarificación de los abonados, el registro y definición de los perfiles de servicio, la gestión y seguridad en el manejo de sus datos, así como la operación de los elementos de la red.

LTE (*Long Term Evolution*), conocida como 4G, es una tecnología de radio plataforma que permite a los operadores a obtener mayores picos de rendimiento que HSPA en un ancho de banda mayor. LTE tiene tres características clave: permite altas tasas de bits con baja latencia, es barato y fácil de desplegar por los operadores, y evita la fragmentación por el tipo de duplexación. LTE no gestiona SMS o llamadas con conmutación de circuitos; de eso se seguirán encargando las redes GSM y demás, con la consiguiente optimización de los costos en infraestructura. Asume una infraestructura de red IP completa y está designada para soportar voz en un dominio de paquetes (Figura 1).

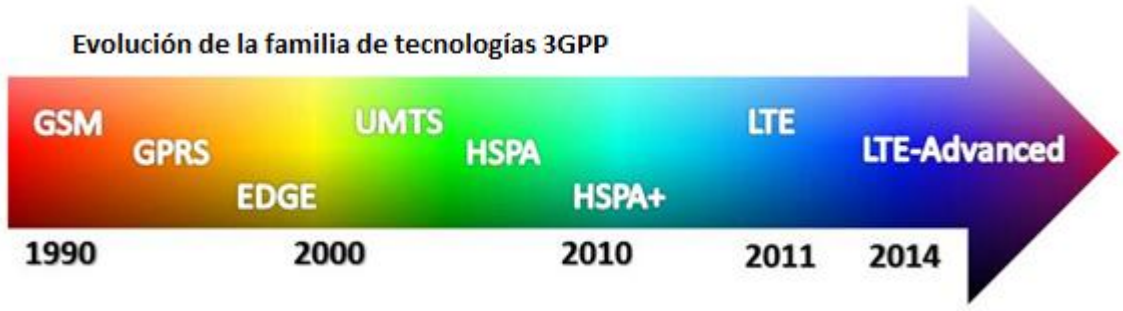

Figura 1. Evolución tecnología celular.

Usuarios de telefonía celular en México

La telefonía celular dejó de ser un servicio exclusivo para un cierto sector y al día de hoy muchos personas son usuarios intensos de esta tecnología y de los servicios, tales como el SMS, correo electrónico, envío de tonos, timbre, imágenes, fotografías, etc.

Al cierre del primer trimestre del 2015, el mercado mexicano contaba con prácticamente 104 millones de líneas de telefonía móvil, cifra cercana a la del mismo periodo del 2014, aunque con una mayor penetración de Internet móvil (Figura 2).

En cuanto a la distribución del mercado entre operadores, prácticamente 7 de cada 10 líneas son de Telcel (América Móvil) por 2 de Movistar (Telefónica). El resto está repartido entre Iusacell y Nextel México, compañías adquiridas recientemente por AT&T. Además, entre 2014 y 2015 nuevos operadores móviles virtuales (OMV) han iniciado sus operaciones, pero su participación en términos de suscriptores es minoritaria todavía.

Un OMV es en términos simples una empresa de telefonía móvil que no cuenta con licencias de espectro y comercializan capacidad usando las redes de operadores que cuentan con concesiones de frecuencias.

Figura 2. Gráfica de usuarios de líneas celulares en México a través del tiempo.

Teléfonos Smartphone

Se puede decir que un smartphone (del inglés smart: inteligente y phone: teléfono), es un teléfono móvil que te permite llevar a cabo acciones propias de una PDA (Asistente digital personal), más allá de lo fuera de lo común en todos los móviles, es decir, llamadas de voz y SMS.

La potencia de cálculo de un smartphone es comparable a la de un ordenador de escritorio o portátil, además deben de ser capaces de ejecutar un sistema operativo móvil (SO móvil) completo e identificable, este SO para móviles ha de tener su propia plataforma de desarrollo de aplicaciones y permitir que éstas tengan una mejor integración con el software base y el hardware del teléfono.

Los primeros smartphones combinaron funciones de asistente digital personal (PDA) con cámara de fotos y navegador GPS, pero ahora incluyen conexión a internet vía WIFI o red móvil para navegar por la web, videollamadas, visionado de correo electrónico (E-Mail), reproductor multimedia, etc.

El primer smartphone de la historia fue el IBM Simon. Fabricado en 1992 y distribuido por Estados Unidos entre agosto de 1994 y febrero de 1995, tenía un precio de 899 dólares, con una interfaz de usuario ausente de botones físicos y basada totalmente en una pantalla táctil de tipo LCD monocromo. Disponía de texto predictivo, agenda, funciones de SMS, correo electrónico, buscador (beeper), fax y un módem para conexión a internet, estas funciones eran más comunes de una PDA que de un móvil de la época.

Actualmente dentro del mercado de la telefonía móvil la tendencia es la de integrar, por parte de los fabricantes, la tecnología GPS dentro de sus dispositivos. El uso y masificación del GPS está particularmente extendido en los teléfonos móviles smartphone, lo que ha hecho surgir todo un ecosistema de software para este tipo de dispositivos, así como nuevos modelos de negocios que van desde el uso del terminal móvil para la navegación tradicional punto-a-punto, la prestación de los llamados Servicios Basados en la Localización (LBS), hasta aplicaciones que haciendo uso de la realidad aumentada y diversos mapas son capaces de localizar por la calle lo que tengan programado.

Aplicaciones móviles

En épocas de tiempo escaso, se impone una sociedad plug-and-play: tener la información necesaria sin depender de conocimientos extra ni trámites tediosos. Por ello, las aplicaciones móviles se han convertido en el punto focal de las tecnológicas. Tanto como que, prácticamente, son los programadores de aplicaciones los que

presionan a los fabricantes de hardware y de sistemas operativos móviles para que ofrezcan nuevas funciones en sus plataformas.

Los consumidores demandan contenidos a un ritmo cada vez más acelerado y la innovación se apura para mantener el ritmo, ubicando a las aplicaciones en un lugar de destaque.

El problema para los desarrolladores es que los dispositivos no están estandarizados, y cubrir todo el mercado exige multiplicar esfuerzos y desarrollar versiones para las distintas plataformas disponibles, tales como Apple, Android, BlackBerry o Windows Phone. Preparar una aplicación para cada dispositivo, significa un producto diferente con lenguaje distinto.

A su vez, no es sencillo posicionar el producto en un gran mercado virtual. Para los desarrolladores de aplicaciones, los ingresos pueden ser algo que llegue a largo plazo. Acceder a un mercado les da un posicionamiento importante a futuro, pero los ingresos directos pueden no ser muy significativos en primera instancia. Según estimaciones de mercado, el 90% de las descargas son aplicaciones gratuitas y el restante 10% tiene precios que oscilan en torno a 1 dólar americano.

Según un informe de la firma de capital emprendedor KPCB del 2014, este año se tienen alrededor de 1.600 millones de smartphones en el mundo. En cuanto a dispositivos móviles se calcula una suma de 5.600 millones. Sin duda una gran audiencia. Según un reciente estudio de la GSMA para 2020 la cifra de smartphones podría ser de 6.000 millones. Casi todo el mundo parece coincidir en afirmar que la mayor parte de ese crecimiento vendrá por parte de países emergentes como India, China, Indonesia o Brasil. Sus tasas de adopción de smartphones todavía son relativamente bajas en comparación con los países desarrollados.

Los ingresos generados en las diferentes app stores están previstos en 45.400 millones de dólares para el final del 2015. Y esa cifra sería de 76.520 millones para 2017, según previsiones de Statista. Esto va unido a las previsiones de descargas de apps, estimadas en 167.050 millones para finales de 2015 y en 211.310 millones para 2016, también según Statista.

Todos estos números del punto anterior se reflejan también en el uso que se hace de las aplicaciones móviles. Según un estudio de Octubre de 2014 de Comscore, el 52% el tiempo que los consumidores estuvieron online lo hicieron desde un dispositivo móvil, bien un smartphone o una tablet.

Uso de tecnologías en sistemas bancarios

Un banco es una institución de tipo financiero que por un lado administra el dinero que les deja en custodia sus clientes y por el otro utiliza el mismo para prestárselo a otros individuos o empresas aplicándoles un interés; esto es una de las variadas formas que tiene de hacer negocios e ir ampliando el dinero de sus arcas. En tanto, se denomina banca o sistema financiero al conjunto de bancos que conforman la economía de un país determinado.

Los sistemas de información son uno de los activos más destacados de las instituciones financieras contemporáneas y aquellos considerados núcleos resultan especialmente importantes y son un ejemplo claro de capacidades distintivas entre entidades, ya que de ellos depende la facultad de una determinada institución para atraer clientes, prestarles sus servicios y conservarlos. Está demostrado que los bancos que crean esas capacidades distintivas se hallan en disposición de imponerse a la competencia y batir a entidades similares en el largo plazo y en cualquier ciclo económico.

No obstante, la decisión sobre sustituir un sistema de núcleo bancario, y el momento y la forma de llevar a cabo su reemplazo, es posiblemente una de las decisiones más importantes que tenga que tomar el director de tecnología de un banco. El actual entorno económico, sumamente complejo, somete a las entidades financieras a una gran presión por lo que se ven obligadas a efectuar agresivos recortes en materia de costos, al tiempo que se deben preparar para disfrutar de cierta ventaja competitiva cuando el ciclo económico cambie y se oriente de nuevo hacia el crecimiento.

En el actual entorno económico, el alto costo en la evolución y el mantenimiento de dichos sistemas de información con, en general, muchos años de vida útil ya no son sostenibles como lo era en épocas de fuerte crecimiento, así como llegan a suponer un obstáculo a la diferenciación, el crecimiento y la generación de negocio del banco. Junto al costo, el otro gran motivo para aplazar la decisión de cambiar los sistemas de núcleo o core básicos es la posibilidad de que se presenten casos de fraude o de lavado de dinero, además de que se puede comprometer información sensible.

En este sentido, la aparición de paquetes de software de core bancarios, significa que los bancos ya no tienen que crear nuevos sistemas de la nada como hicieron hace décadas cuando desarrollaron sus sistemas básicos, lo que minimiza los riesgos de que la actividad de la empresa se vea alterada, y reduce el tiempo y el costo de las iniciativas de sustitución de estos sistemas básicos.

México es uno de los 148 países miembros del Banco Mundial (BM) donde la población de bajos recursos utiliza menos los servicios financieros formales. Mientras 16% de los adultos de África al sur del Sahara refirió que ha utilizado un teléfono celular para pagar facturas y enviar o recibir dinero en los últimos 12 meses, en

México sólo 6% de los adultos ha utilizado esta opción para mover recursos o recibir alguna transferencia.

De acuerdo con la encuesta anual levantada entre 150,000 personas de 148 países, 41% de los latinoamericanos que no poseen una cuenta en el banco indicó que los costos son la principal razón para no ser clientes de un banco. Sin embargo, 19% de los adultos de la región reconoce que tiene una tarjeta de crédito en comparación con 5% del resto del mundo en desarrollo.

El caso para México es muy similar al de la región. Según el documento “Medición de la inclusión financiera: Base de datos global Findex”, 13% de los adultos consultados refiere que tiene tarjeta de crédito, pero sólo 7% dice tener una cuenta de ahorro en el sector bancario que respalde sus gastos.

De los adultos mexicanos consultados, 8% refirió que recibió un préstamo de una institución financiera formal el año pasado, mientras 15% evidencia que son familiares y amigos sus principales fuentes de financiamiento.

En los 148 países consultados, hay unas 2,500 millones de personas que no están bancarizadas. De acuerdo con el presidente del Banco Mundial, Robert Zoellick, al incluir a estas personas como clientes de los servicios financieros formales habría una doble ventaja. Se podría reforzar el crecimiento económico y las oportunidades de las personas pobres de todo el mundo. Se explica que las herramientas financieras de ahorro, pagos y créditos son una necesidad esencial para las personas pobres, especialmente, las mujeres, y pueden ayudar a las familias y a comunidades a salir de la pobreza.

La mayor parte de la población en América Latina y Caribe carece de acceso a servicios financieros a través de canales formales, especialmente entre la población de rentas medias y bajas. Entre las principales causas de esta circunstancia destacan la baja competencia entre entidades financieras y los altos márgenes de intermediación, que encarecen la oferta financiera y hacen inviable la prestación de servicios a población geográficamente dispersa o con bajos ingresos.

En este punto es destacable la acción de las entidades microfinancieras y, allí donde la regulación lo permite, de agentes o corresponsales no bancarios. Unos y otros están contribuyendo a rebajar el umbral de población no atendida y han inducido una nueva gama de productos accesibles ofrecida por la banca comercial e impulsada por los reguladores financieros.

Instituciones de regulación bancaria

El objetivo principal de la regulación y la supervisión bancarias es evitar una crisis del sistema de pagos de la economía. Por este motivo, la supervisión se debe enfocar a la asignación de los activos financieros (crédito otorgado), el capital requerido para garantizar la solvencia de las instituciones y la constitución de reservas que amparen los riesgos en que incurre cada banco.

En este sentido, la regulación y la supervisión deben tener como mandato central prevenir retiros masivos de depósitos —fruto de la desconfianza— en contra de los bancos que puedan llevarlos a incumplir sus obligaciones, es decir, no cubrir los depósitos de sus clientes. Al ser los depósitos en cuentas de cheques —uno de los pasivos de la banca— el principal medio de transacciones, las instituciones reguladoras fungen como reguladores del sistema de pagos al evitar prácticas bancarias que pudiesen conducir a una crisis generalizada de este sistema.

En el diseño institucional de la regulación y la supervisión se deben adicionar a las entidades encargadas de vigilar la actividad bancaria, mecanismos de mercado que estimulen a los bancos a reducir sus riesgos los cuales permitan disminuir los costos de la supervisión.

La regulación tiene el objetivo de garantizar el buen funcionamiento del mercado y de proteger los intereses de los ciudadanos. Esto implica velar por la estabilidad del sistema financiero (previniendo riesgos sistémicos) y garantizar la seguridad de las transacciones financieras, especialmente cuando existe una captación de ahorro. Sin embargo, también debe dejar espacio para la innovación, como camino para satisfacer mejor las necesidades de los ciudadanos. Además, en una región con las desigualdades económicas de América Latina y el Caribe, la asequibilidad y los objetivos de inclusión financiera deben ser igualmente una prioridad para los reguladores y supervisores.

La Comisión Nacional Bancaria y de Valores (CNBV) tiene por objeto supervisar y regular, en el ámbito de su competencia, a las entidades financieras, a fin de procurar su estabilidad y correcto funcionamiento, así como mantener y fomentar el sano y equilibrado desarrollo del sistema financiero en su conjunto, en protección de los intereses del público. También tiene como finalidad supervisar y regular a las personas físicas y morales, cuando realicen actividades previstas en las leyes relativas al sistema financiero.

De entrada, ningún país latinoamericano prohíbe taxativamente la prestación de servicios financieros a través del celular. Se regula la prestación de servicios financieros a través de agentes o corresponsales no bancarios, que cubren poblaciones en las que no hay presencia de sucursales o cajeros automáticos, por no ser éstos rentables.

La normativa contra el blanqueo de capitales y la financiación del terrorismo exige la identificación detallada de las partes implicadas en una transacción financiera. Es un punto delicado si tenemos en cuenta que más del 80% de las líneas móviles de la región funcionan en modalidad prepago y no requieren la identificación del cliente.

Se debería definir claramente el concepto de dinero electrónico, las entidades que pueden emitirlo, si el servicio constituye una captación de ahorro, qué grado de conocimiento del cliente es necesario y, en su caso, las limitaciones de saldo medio y/o de importe de cada operación

Contexto profesional

En México se necesita mayor cultura financiera, fortaleza en la información y más protección al consumidor, así como la inclusión financiera, pues se estima que 47% de los habitantes de los países en desarrollo no tiene acceso a servicios financieros formales, lo que es un factor de inequidad social. Lo cierto es que con más de 100 millones de contratos de telefonía móvil que hay en el país, según datos de la Comisión Federal de Telecomunicaciones, estos dispositivos tienen una penetración mayor que cualquier canal o producto financiero que la banca haya implementado hasta ahora.

Los servicios financieros ofrecidos a través del teléfono móvil han surgido más tarde en América Latina y el Caribe que en otras regiones emergentes como África o Asia, donde existen servicios comerciales operando desde hace varios años. Las primeras experiencias latinoamericanas son todavía recientes y los reguladores financieros están desarrollando en estos momentos la normativa necesaria para permitir su desarrollo con plenas garantías de estabilidad y seguridad de las transacciones.

Las condiciones de México lo hacen muy apropiado para que se den otros servicios financieros a través del móvil:

- Bajos niveles de bancarización, cobertura limitada y costos elevados.
- Territorio extenso y zonas no atendidas por entidades financieras.
- Principal receptor latinoamericano de remesas.
- Próxima regulación de corresponsales no bancarios, experiencias ya en marcha.
- Penetración móvil camino del 80%.
- Relativa sofisticación de los clientes mexicanos de telefonía móvil (alto uso de SMS y alto porcentaje de ingresos celulares de datos sobre ingresos totales).

Es por eso que se creó este servicio que permite a usuarios de telefonía móvil realizar pagos móviles, transferencias de dinero y consultas en tiempo real, las 24 horas del día, los 7 días de la semana, a través del envío de mensajes de texto (SMS). También se pueden realizar depósitos, retiros de efectivo en cajeros automáticos sin necesidad de tarjeta y recarga de tiempo aire (Figura 3).

Esta plataforma surge de la unión temporal entre el proveedor de telefonía celular en México y bancos importantes del mismo país para transformar el teléfono celular en un innovador y eficiente instrumento de pago para transferencias.

La compañía de servicios integrales de telecomunicaciones en América Latina cuenta alrededor de 242 millones de suscriptores celulares, teniendo operaciones en 18 países. Su unidad local en México, cuenta con más de 67.5 millones de clientes. El

banco socio de este proyecto cuenta con una extensa red de distribución de cerca de 1,662 sucursales, 5787 cajeros automáticos y más de 4,200 corresponsalías ubicadas en todo el país. **Gemalto** es el proveedor tecnológico en México de la plataforma transaccional de pagos móviles y es el responsable del desarrollo, soporte y operación del servicio. El proyecto de pagos móviles garantiza que los clientes puedan utilizar sus cuentas móviles a través de diferentes canales de manera segura.

México es el primer país de América Latina en contar con un servicio de esta naturaleza, aunque el plan de evolución del producto incluye ofrecer los servicios **a otros países en Latinoamérica**. Por el momento, la plataforma funciona sólo entre usuarios de una compañía celular en particular, aunque la plataforma está abierta y la idea es que se sumen otros bancos al ecosistema ya definido, así como la inclusión de otras compañías telefónicas también es una posibilidad.

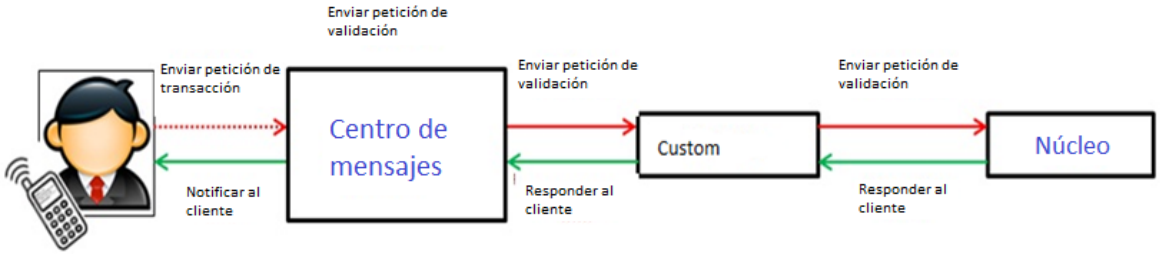

Figura 3. Flujo genérico de transacción por sistema de pagos móviles

Capítulo 1: Organigrama

GGS (Servicios Globales de Gemalto) propone su conocimiento en la construcción de servicios de alto valor para soluciones personalizadas basadas en tarjetas. Desde el diseño hasta la implementación en ambientes operativos, GGS asegura servicio continuo para todo tipo de proyectos.

Con más de 600 aplicaciones y soluciones móviles implementadas alrededor del mundo, se ayuda a operadores a construir su estrategia de negocios al proveer soluciones wireless hechas a la medida. Para completar su rango de productos, Gemalto está dispuesto a trabajar con socios locales para proporcionar soluciones apropiadas para el mercado regional. Gracias a la gran experiencia en soluciones punto a punto en telecomunicaciones, GGS facilita experiencia técnica para optimizar e implementar servicios redituables. Todos los equipos están administrando proyectos de clientes de acuerdo a un proceso de calidad certificado.

De manera alternativa a implementaciones en sitio en las instalaciones de los operadores, GGS también puede administrar parte o todos los servicios en nombre de sus clientes, resultando en el incremento del portafolio de servicios y nueva generación de ingresos, a su vez limitando la inversión de recursos e infraestructura.

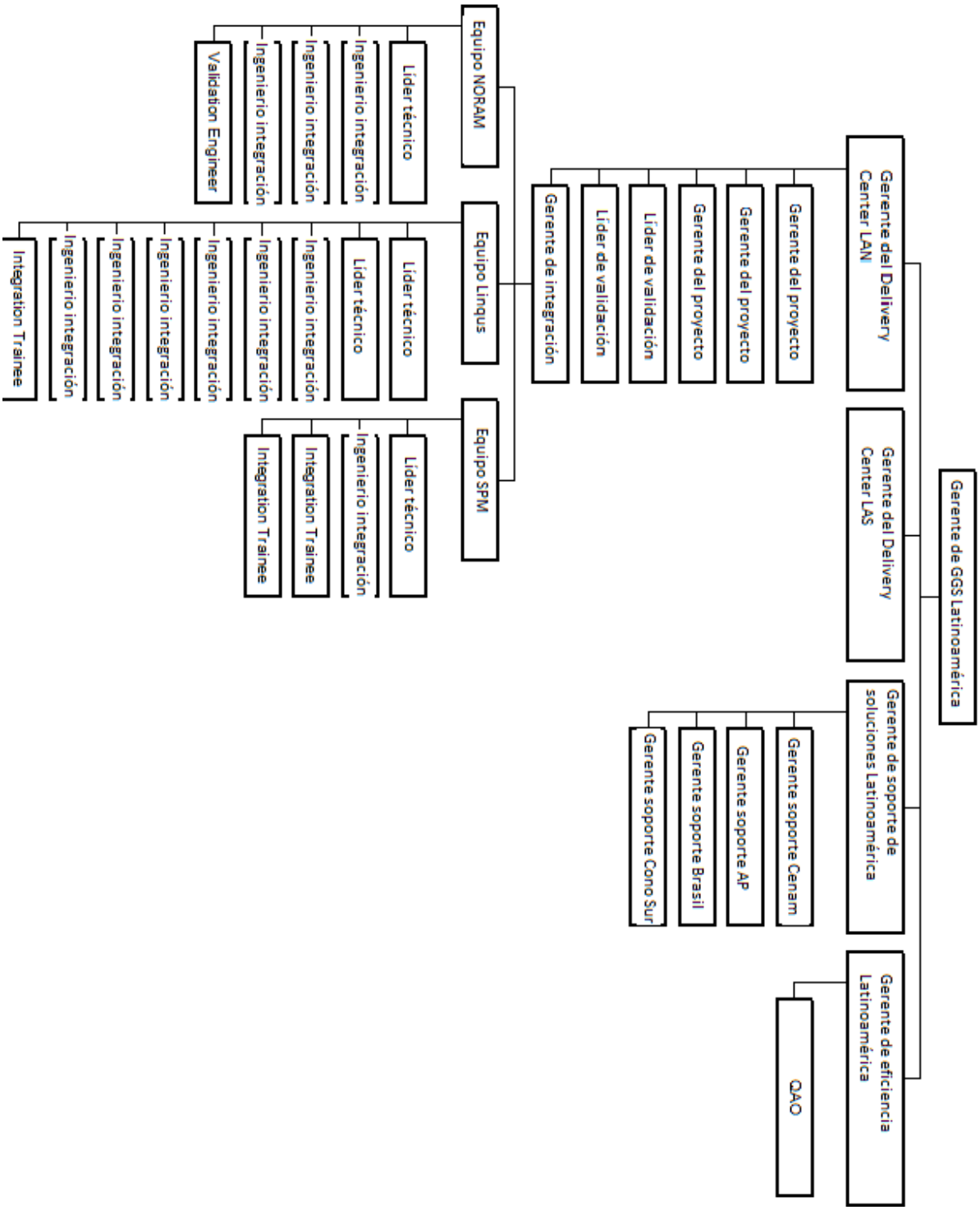

Figura 4. Organigrama GGS Gemalto

Capítulo 2: Descripción de proyectos

Dentro de las soluciones generadas por Gemalto, se utiliza una metodología estándar conocida como CDLC (Ciclo de Vida del Desarrollo del Cliente). EL CDLC es un framework genérico que puede ser modificado dependiendo de cada proyecto; estos cambios son gestionados por el administrador del proyecto

Las fases básicas del CDLC se ejemplifican con el siguiente esquema (Figura 5):

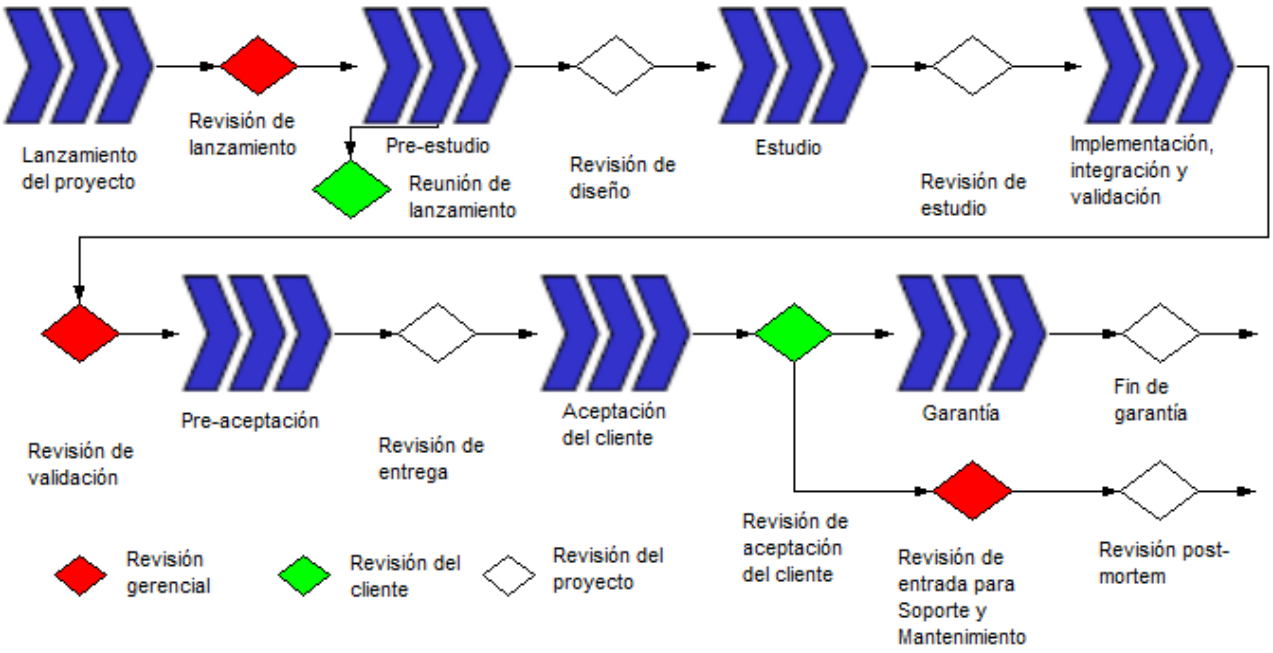

Figura 5. Diagrama metodología CDLC

Los roles establecidos en el CDLC para proyectos estándar son los siguientes:

- Equipo de venta de soluciones
- Gerente de ventas
- Consultor técnico
- Experto en soluciones
- Representantes gerenciales de GGS (Gerente del programa)
- PM (Gerente del proyecto)
- Líder técnico de desarrollo
- Líder técnico de integración
- Equipo de proyecto
- Experto técnico
- CMO
- QAO

- Gobernanza técnica
- Trainer
- Soporte

La primera fase se conoce como lanzamiento del proyecto. El objetivo en esta fase es preparar la transferencia del proyecto, con el fin de que la propiedad pase del equipo de ventas al equipo del proyecto. Se finaliza la base del proyecto y se actualiza la cotización de ser necesario.

Se requiere concentrar toda la información de la etapa de pre-ventas: El criterio de satisfacción del cliente, la propuesta técnica y comercial de Gemalto, las propuestas de los subcontratados, riesgos identificados, suposiciones y explicaciones de la cotización, el contexto comercial y las expectativas específicas de los clientes.

Una vez que se tiene la propuesta comercial y técnica y el CRS (Especificación de requerimientos del cliente), se deben validar costos de hardware, outsourcing y training interno e identificar riesgos técnicos/no técnicos. A partir de este análisis se define el equipo con roles y responsabilidades definidas, calendario maestro, presupuesto y entregables esperados, dentro de los cuales se encuentra el SOW (Statement of Work – Declaración de trabajo); este documento tiene como propósito ser una definición base que sirve para entender el funcionamiento básico, módulos, los procesos y las entidades fundamentales que intervendrán en el sistema.

La siguiente etapa se le conoce como **Pre-estudio**. Los objetivos de esta etapa son finalizar las especificaciones de requerimientos de los clientes y actualizar el plan de ejecución del proyecto. Una vez que se cuenta con el SOW, se debe revisar con el cliente para obtener su autorización; se busca confirmar la factibilidad técnica del proyecto, se inicia el proceso de adquisición de hardware, software y outsourcing; además que se debe explicar la configuración necesaria del Hardware y Software en un documento llamado PDP (Plan de despliegue del proyecto). El objetivo es que el cliente (en este caso los bancos y la compañía proveedora de telefonía celular) esté de acuerdo en que el SOW se adecua a lo que se manejó en el CRS y dé el visto bueno. También se empieza a trabajar en la definición de criterios de aceptación y se compromete con un plan de proyecto. En esta etapa también comienza la creación del Plan de políticas de pruebas (TPP) y el Plan de aceptación de pruebas (ATP).

Hablando específicamente de la adquisición de hardware y software, se deben seguir los siguientes pasos: confirmación del GO para la adquisición de HW y SW de terceros por parte del gerente del proyecto, validación y expedición interna de la requisición, validar la requisición de la compra, seguimiento de la orden de compra e involucramiento del equipo de Technical Governance (gobernancia técnica).

Se lleva a cabo una reunión con el cliente conocida como reunión de Kick Off o inicio, donde se presenta a los representantes de los clientes la forma en la que se tiene

planeado alcanzar los objetivos del proyecto, así como obtener el compromiso del cliente acerca de los entregables esperados de su parte, si es que es el caso.

Para pasar a la siguiente fase, se lleva a cabo una reunión interna conocida como Revisión Interna de diseño, donde se asegura que todas las post-condiciones de la fase previa han sido cubiertas, se asegura que todos los miembros del proyecto tienen el mismo entendimiento de los requerimientos funcionales, la estrategia de pruebas, el criterio de aceptación, las tareas de despliegue o deployment, todo en línea con lo descrito en el SOW. Así como todos deben estar comprometidos en la factibilidad técnica, el calendario y los criterios de aceptación y de calidad.

En la fase de **Estudio** los dos principales propósitos son definir el diseño del software y detallar la implementación y la estrategia de validación. Se detalla las funcionalidades del software y se define el diseño (arquitectura, componentes principales e interfaces); se detalla la estrategia e implementación de pruebas; se asegura la trazabilidad de los requerimientos por medio de la documentación de validación, diseño y especificación; se completa la estrategia de pruebas con la estrategia de integración; se definen las pruebas a ser ejecutadas; así como revisar que todas las actividades en esta etapa son consistentes con los niveles de seguridad requeridos y con los resultados de los análisis de seguridad. Se actualiza la estrategia de desarrollo, se define un ambiente de desarrollo e implementación, se actualiza el plan de aceptación, además de mantener un seguimiento con el cliente.

Con respecto a la documentación, se define el plan de pruebas de integración, la descripción de diseño de software y el plan de pruebas de validación (si es que aplica); a su vez se actualiza el plan de políticas de pruebas y el plan de despliegue del proyecto. De manera similar a etapas anteriores, se convoca a una junta (revisión de estudio) para determinar si todas las post-condiciones fueron alcanzadas y así poder dar el GO para proceder a la siguiente etapa.

La siguiente etapa es la de **implementación, integración y validación**. Los objetivos primordiales son implementar y probar el software para comprobar si cumple lo establecido en el SRS (Especificación de Requerimientos de Software) y probar la solución como un todo en un ambiente tan parecido como sea posible al ambiente objetivo (el ambiente productivo). Dentro de las actividades que se llevan a cabo en esta etapa está detallar especificaciones de diseño con alto grado de refinamiento, configurar y mantener el ambiente de desarrollo (hardware y software), asegurar la implementación de la arquitectura del software, instalar los componentes de hardware y software necesarios, generar pruebas de unidad de código, integrar componentes de software, hacer pruebas para detectar y documentar defectos, y crear guías de usuario final y materiales de entrenamiento, completar la documentación técnica (guías de administración y guías de instalación) y escribir la OMG (Guía de operación y mantenimiento). Todo esto con el propósito de verificar que la solución probada se apegue a lo que el cliente definió en el documento de especificaciones (CRS).

En esta etapa el equipo de oficiales de administración de configuraciones (CMO) atiende peticiones y brinda soporte y dirección en el manejo de herramientas para administrar repositorios (SVN) y tracking de defectos (Jira y Quality Center).

La Revisión de Validación es mandatoria y su objetivo es que el Project Manager y el equipo de integración verifiquen los componentes entregados (principalmente bugs abiertos y escenarios no probados) y autoricen la entrega interna. En particular, el equipo de integración debe hacer la confrontación con respecto a la validación e integración, así como comunicar los riesgos identificados yendo hacia adelante.

El objetivo en la etapa de **Pre-aceptación** es preparar y verificar los elementos para la aceptación del cliente. Se le entrega al cliente el ATP (Plan de aceptación de pruebas) con el fin de que sea aprobado por ellos. El ATP es un documento que describe detalladamente la estrategia de pruebas que se seguirá para validar la calidad del producto antes de su lanzamiento; describe el alcance, los recursos y el calendario a seguir durante el periodo de pruebas. En esta etapa también se finaliza el despliegue de la solución, se realizan pruebas internas antes de presentar al cliente y se realiza un training para el cliente si es necesario. Además, se comienza a preparar el handover o traspaso para el equipo de operaciones.

En este caso, existe un equipo de QA (Aseguramiento de calidad) para probar las piezas de software desarrolladas; los componentes se instalan en un ambiente separado al que se usará en etapas de pruebas posteriores. El equipo de QA se apega a una matriz de pruebas específica y registra y documenta los errores que se encuentren para mandarlos al equipo de desarrollo para su resolución.

Se lleva a cabo la Revisión de Entrega para establecer si la entrega completa está lista y cumple con los requerimientos contractuales necesarios antes de entregarla para la aceptación externa. Como input se requiere el ATP firmado y aprobado por los clientes, así como el Reporte de pruebas de validación y el Reporte de Lista de anomalías obtenido de las pruebas internas.

Completado el proceso anterior se inicia con la **aceptación**, en la cual se entrega la solución al cliente para probar que los entregables se apegan a lo que fue solicitado en los requerimientos del cliente.

Esta etapa de pruebas también se conoce como UAT (pruebas de aceptación de usuarios). En este caso se instala una versión del sistema con sus respectivos customs en un ambiente independiente para que el cliente pruebe todos los casos de usos correspondientes y se solucionen los bugs o anomalías que surjan del esquema de pruebas. Los managers se aseguran que las pruebas realizadas se apeguen a lo establecido en el ATP y los integradores son los encargados en proveer soporte a los clientes en caso de que se requiera.

La Revisión de Aceptación del Cliente es una clausura formal con los clientes, en orden de verificar que todo ha sido cerrado (es decir, no hay puntos pendientes de ninguna de las dos partes), confirmar su aceptación (con el ATP firmado) y finalmente introducir al equipo de soporte y mantenimiento que actuará desde este punto en adelante como el punto focal del proyecto. Sin importar la decisión tomada en esta reunión, ésta debe ser informada al gerente de ventas y al equipo de ventas.

La Revisión de entrada para Soporte y Mantenimiento (conocida como SAMIR) es una reunión entre los equipos de delivery o entrega y de Soporte con el fin de asegurar que todos los elementos necesarios están disponibles para la etapa de soporte y mantenimiento, y también validar la transferencia de responsabilidades del equipo del proyecto al equipo de soporte.

Los inputs necesarios para esta revisión son los siguientes:

- Estatus de la Revisión de Aceptación del Cliente
- Fecha de inicio y fin de la garantía
- Documentos almacenados en el repositorio propiedad del equipo de soporte (Guía de operación y mantenimiento, documento de acceso remoto, entre otros)
- Lista de los componentes del producto a ser modificados
- Solución de backup/restore para cada plataforma
- Información de los contratos de elementos provistos por terceros (inicio y fin de contratos, contactos locales, nivel de servicio, referencia de productos y números de serie)
- Casos de uso
- Estatus del actual contrato de soporte

El equipo de entrega debe entrenar al equipo de soporte con respecto a las customizaciones específicas del cliente y la información específica de la operación y mantenimiento del proyecto. El objetivo es que el equipo de operaciones conozca todas las características de la solución y tener un buen entendimiento de los ítems y customs del proyecto.

La aceptación del SAMIR depende de la existencia de puntos bloqueantes. Se le considera un punto bloqueante a elementos o información crítica que tengan un fuerte impacto en la capacidad de realizar las actividades de operación y mantenimiento. Ejemplo de estos casos son: conexión remota no establecida, guía de operación y mantenimiento no creada o actualizada, falta de información acerca de los requerimientos del cliente, falta de información de los compromisos establecidos con el cliente, falta de información de troubleshooting o resolución de problemas, entre otros. En caso de que exista un punto bloqueante, el gerente del proyecto debe planear un nuevo SAMIR una vez que se hayan resuelto los puntos pendientes.

En caso de que se dé el Go (aprobación) por parte del equipo de soporte y operación, se pasa a la etapa de **garantía**. El propósito de la etapa de garantía es asegurar la solución de bugs y cambios de requerimientos mientras el control del proyecto pasa progresivamente al equipo de operación y soporte. El soporte realizado durante esta etapa recae en los integradores. El responsable del proyecto sigue siendo el equipo de entrega; el único cambio es que el equipo de soporte funge como la interfaz de cara al cliente. Una vez pasado el periodo de garantía la responsabilidad total del proyecto recaerá en el equipo de soporte. Cuando termina esta etapa, por lo general se firma un contrato de soporte; esto significa que los ingenieros de soporte trabajan en un proyecto diferente de manera interna.

La reunión Post Mortem tiene como propósito cerrar el proyecto y validar la transferencia de responsabilidad del equipo de entrega al equipo de soporte. Los objetivos son recopilar lecciones aprendidas y ponerse de acuerdo en cómo mejorar en el futuro; en algunos casos también se lleva a cabo una reunión similar con los clientes si es requerido.

El periodo de garantía dura por lo general 3 meses. Se debe informar tanto a los cliente como de manera interna que el mantenimiento será realizado totalmente por el equipo de soporte. Se cierra el proyecto de entrega y la responsabilidad del SLA pasa al manager o SDM encargado de soporte.

En la etapa de mantenimiento (soporte y operación), el cliente y el proveedor firman lo que se conoce como SLA (Arreglo de nivel de servicio). Un SLA es un protocolo plasmado normalmente en un documento de carácter legal por el que una compañía que presta un servicio a otra se compromete a prestar el mismo bajo unas determinadas condiciones y con unas prestaciones mínimas. El nivel de servicio se basa en indicadores que permiten cuantificar de manera objetiva determinados aspectos del servicio prestado.

Un ejemplo de indicador de nivel de servicio en este caso es el tiempo de resolución de incidencias. Este indicador se mide a través de aplicaciones de gestión de incidencias que registran el momento que una incidencia es comunicada y cuándo es cerrada. La diferencia entre estos dos datos es el indicador en bruto desagregado que luego puede ser procesado mediante algoritmos para obtener promedios, desviaciones y otros indicadores normalizados.

En un SLA se pueden establecer tantos indicadores como se estime necesario y de su evaluación se obtienen por ejemplo penalizaciones a la empresa suministradora, identificación de puntos débiles del proceso e indicaciones para procesos de mejora continua (CMM) en determinadas actividades.
CDLC es el framework utilizado para los proyectos de Gemalto (Figura 6)

- Soportado por procesos verticales

- Define una lista de fases con sus propios objetivos
- Define revisiones para asegurar conclusiones exitosas de las fases:
    - Revisiones de proyectos para compartir información y tomar acciones dentro del equipo del proyecto
    - Revisión gerencial para el estatus de GO/NO GO
    - Revisión del cliente para la aceptación
- Cada revisión es definida por sus inputs y criterios de aceptación
- CDLC puede ser customizado para atender características del proyecto.

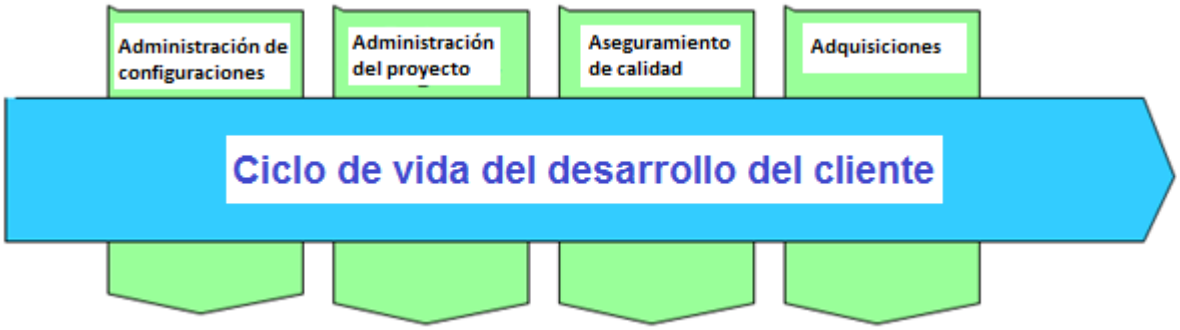

Figura 6. Framework CDLDC

Capítulo 3: Proyecto principal (Implementación genérica de Sistema de pagos móviles)

La solución provee una plataforma transaccional que permite a los operadores ofrecer un amplio rango de servicios financieros móviles. Los clientes serán capaces de extender servicios a los suscriptores, creando un nuevo flujo de ingresos, fortaleciendo las marcas y presentando un modelo innovador y creativo. Los clientes se beneficiarán de la conveniencia de realizar transacciones financieras de rutina desde su teléfono celular y varios medios de acceso adicionales.

Instalación

Para la etapa de pruebas del sistema de pagos móviles, es necesario implementar un ambiente exclusivamente enfocado en realizar las pruebas. El primer paso es elegir un servidor con los requerimientos necesarios de memoria y de almacenamiento para poder soportar todos los componentes a ser implementados. En este caso se sugiere utilizar un servidor RedHat.

RedHat es posiblemente la distribución más popular de Linux disponible para el mercado del servidor. Sus características son**:**

- Proporciona la instalación fácil y las herramientas de gran alcance de la administración para los usuarios.
- Ayuda extendida del hardware para todas las configuraciones de sistema.
- Instalaciones gráficas realzadas.
- Estructura superior de los archivos y capacidades avanzadas de los trabajos múltiple.
- Ofertas eficientes y sistema-RPM de empaquetado bien integrada.
- Memoria y estabilidad avanzadas en la ejecución de programa.
- Contiene las características extensas de la seguridad del sistema para los usuarios.

Red Hat Enterprise Linux (RHEL) es un sistema operativo basado en Linux diseñado para negocios. RHEL puede trabajar para en desktops, servidores, hipervisores o en la nube. RHEL tiene múltiples variables, con versiones servidor para x86, x86-64, Power PC, Itanium, entre otras.

Siendo una distribución Linux, RHEL contiene el kernel de Linux, así como también algunas aplicaciones para realizar ciertas tareas. Como todas las distribuciones de Linux, RHEL es open source. De esta manera, la gente puede observar su código fuente y crear sus propias versiones customizadas.

Una vez que el servidor es montado exitosamente, el siguiente paso es instalar la instancia del sistema. En esta etapa es necesaria instalar 2 elementos: el sistema gestor de base de datos y el servidor de aplicaciones.

En el caso de la base de datos, se prevé implementar Oracle Database; es un sistema de gestión de base de datos objeto-relacional, desarrollado por Oracle Corporation. Se considera a Oracle Database como uno de los sistemas de bases de datos más completos, destacando: soporte de transacciones, estabilidad, escalabilidad y soporte multiplataforma. Las últimas versiones de Oracle han sido certificadas para poder trabajar bajo GNU/Linux.

Un RDBMS es un Sistema Gestor de Bases de Datos Relacionales. Se trata de software capaz de producir, manipular y gestionar bases de datos de tipo relacional. Es un software que se antepone a los datos de una base de datos, de modo que cualquier acceso a los datos pasa por una petición al RDBMS que éste gestiona a fin de realizar la operación más conveniente sobre esa petición.

La gran ventaja de los RDBMS consiste en que permiten gestionar los datos de forma lógica, se utilizan estructuras más abstractas para los datos, a fin de evitar utilizar el complicado entramado físico que posee una base de datos. El diccionario de datos agrupa los metadatos de una base de datos. En este diccionario aparecen todos los objetos de la base de datos; con su nombre, función, control de acceso (seguridad) y correspondencia física en los archivos de datos.

Oracle aprovecha un nuevo enfoque, el objeto-relacional. Es un punto medio entre los modelos relacionales y orientados a objetos, permite a los usuarios utilizar la base de datos de forma relacional, pero incorpora extensiones de las bases de datos orientadas a objetos. Oracle incluso soporta el enfoque orientado a objetos. El hecho de permitir los tres objetos hace que se pueda trabajar de forma relacional y añadir cualquier mejora orientada a objetos. Pero el núcleo de Oracle sigue estando pensado para el enfoque relacional.

Un servidor Oracle es el software que permite una administración y desarrollo de bases de datos. Tiene tres posibilidades de ejecución:

- Local o basada en host: El servidor se ejecuta en la misma máquina en la que se conectan los clientes. La versión personal de Oracle database, produce servidores de este tipo.
- Cliente-Servidor: Enfoque más típico. El servidor reside en un ordenador distinto respecto al que los usuarios van a usar para conectarse a la base de datos.
- Cliente-Servidor de Aplicaciones-Servidor: Los usuarios acceden a un servidor de aplicaciones (Oracle Application Server) que, a su vez, accede al servidor Oracle. Los tres elementos (cliente, servidor de aplicaciones, servidor Oracle) pueden estar en tres máquinas distintas.

El servidor Oracle está formado por dos elementos:
La instancia de la base de datos: Consta de datos (llamados estructuras de memoria) y de procesos en memoria (procesos background) necesarios para dar servicio a los usuarios de la base de datos. Puede haber más de una instancia si se distribuye la base de datos en más de una máquina. Cada instancia abre una y sólo una base de datos.
Ficheros en disco: Representan la base de datos en sí. Consta de:

- Estructuras lógicas: Tablespaces, objetos del esquema de usuario.
- Estructuras físicas: Los ficheros de datos almacenados en disco. Los ficheros de datos (asociados a los tablespaces), los ficheros redo log y los ficheros de control (Figura 7).

Para establecer una sesión con la base de datos, el usuario necesita conectar con la instancia de la base de datos. Normalmente esto significa arrancar una herramienta cliente como SQL*Plus o ejecutar una aplicación de desarrollo de bases de datos (como Oracle Forms); entonces se ejecuta un proceso de usuario. Cuando esto ocurre, en el servidor se establece un proceso de servidor. Este proceso es el encargado de comunicar al usuario con la instancia Oracle en nombre del proceso de usuario. Cada vez que el usuario ejecuta instrucciones SQL, éstas son transmitidas a la instancia Oracle por el proceso servidor.

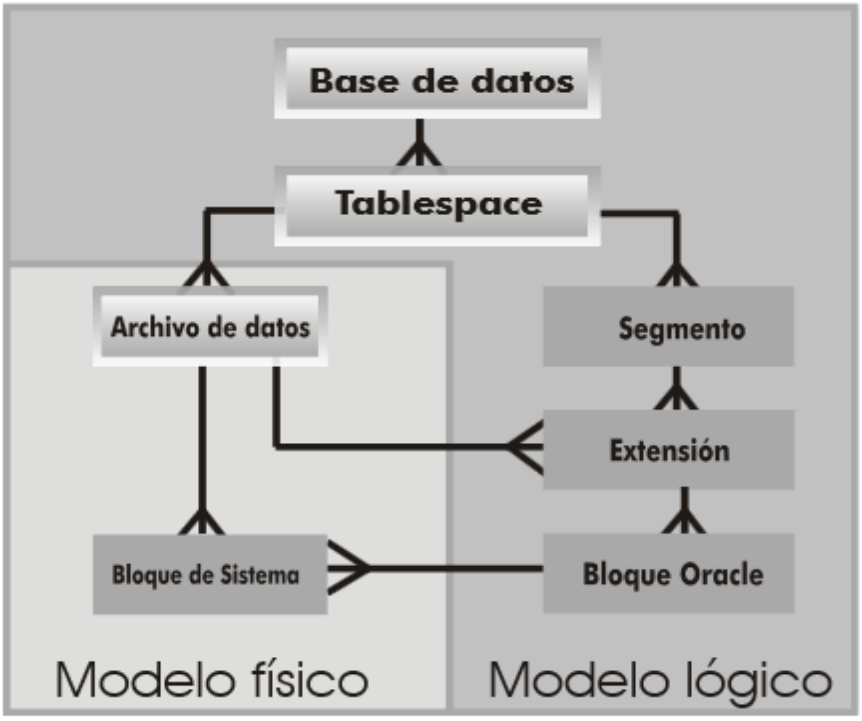

Figura 7. Modelo físico y lógico de base de datos

Para poder ejecutar SQL sobre la base de datos, hay que conectar con la instancia Oracle de la base de datos, lo cual requiere la comunicación entre un proceso cliente y el servidor (el proceso cliente puede ser una instancia de SQL*Plus por ejemplo). Una transacción son varias operaciones SQL que forman una unidad de trabajo. Comienza cuando una persona se conecta y de ahí hasta que ejecuta la instrucción commit (ejecutar la transacción) o rollback (anular la transacción). La anulación deja la base de datos en el estado anterior al comienzo de la transacción. Tras un commit o un rollback comienza la siguiente transacción.

Instalación Oracle 11g en RHEL

1. Establecer los permisos necesarios

 # chown -R oracle.oinstall /home/oracle
 # /usr/sbin/usermod -g oinstall – G dba,oper oracle
 # chown -R oracle:install /home/oracle/database
#chmod -R 775 /home/oracle/database

2. Determinar el shell del usuario oracle:

#su – oracle
$ echo $SHELL
$ exit

Se ejecuta

mkdir -p /u01/app/oracle
chown -R oracle:oinstall /u01/app
chmod -R 775 /u01/app

3. Configurar los parámetros del Kernel

Oracle Database 11*g* requiere el establecimientos de los valores de los parámetros del kernel tal y como se muestran a continuación. Los valores mostrados son mínimos, si el sistema tiene valores superiores no se deben cambiar. Linux permite la modificación de muchos parámetros del kernel si necesidad de reiniciar el sistema.

Editar sysctl.conf

kernel.shmall = 2097152
kernel.shmmax = 536870912
kernel.shmmni = 4096
kernel.sem = 250 32000 100 128
fs.file-max = 65536
net.ipv4.ip_local_port_range = 1024 65000

net.core.rmem_default=4194304
net.core.wmem_default=262144
net.core.rmem_max=4194304
net.core.wmem_max=262144

4. Estableciendo los límites del shell para el usuario Oracle

Para la mejora del desempeño existen una serie de ajustes de los valores límites del shell, que se indican a continuación:

```
cat >> /etc/security/limits.conf <<EOF
oracle soft nproc 2047
oracle hard nproc 16384
oracle soft nofile 1024
oracle hard nofile 65536
EOF

cat >> /etc/pam.d/login <<EOF
session required /lib/security/pam_limits.so
EOF
```

Cambiamos el profile por defecto para bash y ksh, así como el login script por defecto para cshell.

```
cat >> /etc/profile <<EOF
if [ \$USER = "oracle" ]; then
if [ \$SHELL = "/bin/ksh" ]; then
ulimit -p 16384
ulimit -n 65536
else
ulimit -u 16384 -n 65536
fi
umask 022
fi
EOF

cat >> /etc/csh.login <<EOF
if ( \$USER == “oracle” ) then
limit maxproc 16384
limit descriptors 65536
umask 022
endif
EOF
```

5. Configurar las variables de Entorno del usuario Oracle

Loguearse con el usuario oracle.

 # su – oracle
$ vi /home/oracle/.bash_profile

Agregar o editar la siguiente línea:

umask 022

Si las variables del entorno ORACLE_SID, ORACLE_HOME u ORACLE_BASE se encuentran y contienen valores entonces se agregan o modifican:

$ ORACLE_BASE = /U01/app/oracle
$ ORACLE_SID=BASEDEDATOS
$ export ORACLE_BASE ORACLE_SID
$ ORACLE_HOSTNAME=database.iss.es
$ export ORACLE_HOSTNAME

Antes de iniciar la instalación quitar los valores de ORACLE_HOME y TNS_ADMIN para que runInstaller ponga los propios:

$ unset ORACLE_HOME
$ unsetTNS_ADMIN

Verificar que el entorno esté correcto con los siguientes comandos:

$ umask
$ env | more

6. Crear los grupos y usuarios de Oracle

El siguiente paso es crear los grupos de linux y las cuentas de usuario necesarias para la instalación y mantenimiento de oracle 11g. Como root, ejecutar lo siguiente:

Verificar si existen:

 # grep oinstall /etc/group
 # grep dba /etc/group

En caso de que no existan, se crean:

 # /usr/sbin/groupadd oinstall
 # /usr/sbin/groupadd dba
 # /usr/sbin/useradd -m -g oinstall -G dba oracle
 # id oracle
uid=501(oracle) gid=502(oinstall) groups=502(oinstall),503(dba)

Establecer el password en la cuenta oracle

passwd oracle
Ex:
 # passwd oracle
Changing password for user oracle.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.

7. Instalar Oracle

Crear un directorio que acoja el software descargado (por ejemplo linux.x64_11gR1_database_1013.zip)

mkdir 11g_db
cd 11gR1_db
unzip linux.x64_11gR1_database_1013.zip
$ cd /11g_db/database

Arrancar el Oracle Universal Installer con la cuenta oracle.

$ ./runInstaller

Durante el proceso de instalación de la base de datos (es decir, las tablas y los usuarios), se deben establecer las variables necesarias: el nombre del host o IP del servidor donde se está instalando, el puerto TCP a utilizar, el nombre de la instancia de la BD (SID) y el password del usuario 'SYSTEM' (el usuario DBA por default). Una vez determinados estos valores, se debe especificar el usuario y password de la base de datos asociada a la aplicación. Finalmente, con el instalador se crean las tablas de la base de datos usando los mismos parámetros mencionados anteriormente.

Un servidor de aplicaciones es un software que proporciona aplicaciones a los equipos o dispositivos cliente, por lo general a través de Internet y utilizando el protocolo http. Los servidores de aplicación se distinguen de los servidores web por el uso extensivo del contenido dinámico y por su frecuente integración con bases de datos (Figura 8). Se decidió implementar Oracle WebLogic**;** es un servidor de aplicaciones Java EE y también un servidor web HTTP desarrollado por BEA Systems posteriormente adquirida por Oracle Corporation. Se ejecuta en Unix, Linux, Microsoft Windows, y otras plataformas.

Entre estas características están:

- *Estándares de Liderazgo*: Soporte Comprensivo de Java Enterprise para facilitar la implementación y despliegue de componentes de aplicación. WebLogic Server es el primer servidor de aplicaciones independientes desarrollado en Java en conseguir la certificación J2EE.

- *Ricas Opciones de Cliente*: WebLogic Server soporta navegadores Web y otros clientes que usen HTTP; los clientes Java que usan RMI (Invocación Remota del Método) o IIOP (Protocolo de Internet Inter-ORB); y dispositivos móviles que usan WAP (Protocolo de Acceso Wireless). Los conectores de BEA y otras compañías permiten virtualmente a cualquier cliente o aplicación legal trabajar con el Servidor de aplicaciones WebLogic.
- *Escalabilidad*: Los recursos críticos se usan eficientemente y la alta disponibilidad está asegurada a través del uso de componentes Enterprise JavaBean y mecanismos como el "clustering" de WebLogic Server para las páginas Web dinámicas, y los almacenes de recursos y las conexiones compartidas.
- *Administración Robusta*: WebLogic Server ofrece una Consola de Administración basada en Web para configurar y monitorizar los servicios del WebLogic Server. Se hace conveniente para la configuración una interfaz de línea de comandos para administrar el servidor WebLogic con scripts.
- *Seguridad*: WebLogic Server proporciona soporte de SSL para encriptar datos transmitidos a través de WebLogic Server, los clientes y los otros servidores. La seguridad de WebLogic permite la autentificación y autorización del usuario para todos los servicios de WebLogic Server. Los almacenes de seguridad externa, como servidores LDAP (Lightweight Directory Access Protocol) puede adaptarse para WebLogic, nos permiten una sola autentificación para empresas. La Interfaz de proveedor de servicios de seguridad hace posible extender los servicios de seguridad de WebLogic e implementar estas características en aplicaciones.
- *Máxima flexibilidad en el desarrollo y despliegue*: WebLogic Server proporciona una estrecha integración y soporte con las bases de datos más importantes, herramientas de desarrollo y otros entornos.

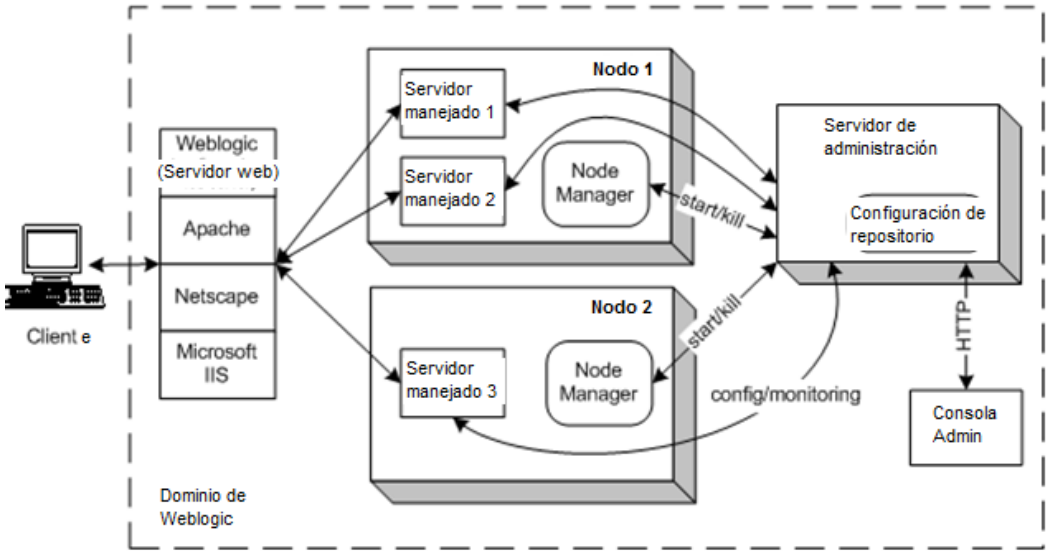

Figura 8. Arquitectura estándar Weblogic

Como prerrequesito para implementar este servidor de aplicaciones se requiere lo siguiente:

- Memoria: 256Mb mínimo (512Mb aconsejable)
- Espacio en disco: 400Mb
- Versión de Java JDK 1.4.1 (se instala junto con el servidor) o superior.

Además de estos prerrequisitos, se deben cargas las variables de entorno utilizando los scripts especificados por default y verificar que ningún otro proceso del servidor de aplicaciones esté corriendo dentro del servidor para la nueva instancia.

JDK es la implementación de las plataformas Java SE, Java EE o Java ME liberada por Oracle Corporation en la forma de un producto binario. Con respecto a la memoria, se debería establecer un mínimo de 1024MB y un máximo de 2048 MB.

Instalación JDK en RHEL

1. Se selecciona el paquete rpm de JDK (jdk-7u25-linux-i586.rpm, jdk-7u25-linux-x64.rpm, jre-7u25-linux-i586.rpm or jre-7u25-linux-x64.rpm).
2. Cambiar a usuario root
3. Instalar el paquete JDK
   ## JDK 32-bit ##
   rpm -Uvh /path/to/binary/jdk-7u25-linux-i586.rpm
4. Instalar java, javac, javaws del JDK

   ## java ##
   alternatives --install /usr/bin/java java /usr/java/latest/jre/bin/java 20000
   ## javaws ##
   alternatives --install /usr/bin/javaws javaws /usr/java/latest/jre/bin/javaws 2000
   ## Java Browser (Mozilla) Plugin 32-bit ##
   alternatives --install /usr/lib/mozilla/plugins/libjavaplugin.so libjavaplugin.so /usr/java/latest/jre/lib/i386/libnpjp2.so 20000
   ## Java Browser (Mozilla) Plugin 64-bit ##
   alternatives --install /usr/lib64/mozilla/plugins/libjavaplugin.so libjavaplugin.so.x86_64 /usr/java/latest/jre/lib/amd64/libnpjp2.so 20000
   ## Install javac only if you installed JDK (Java Development Kit) package ##
   alternatives --install /usr/bin/javac javac /usr/java/latest/bin/javac 20000
   alternatives --install /usr/bin/jar jar /usr/java/latest/bin/jar 20000

5. Verificar versiones de java, javc, javaws

   java -version

java version "1.7.0_25"
Java(TM) SE Runtime Environment (build 1.7.0_25-b04)
Java HotSpot(TM) 64-Bit Server VM (build 22.1-b02, mixed mode)

javaws
Java(TM) Web Start 10.0.0.4-fcs
[...]

javac -version
javac 1.7.0_25

6. Cambiar la versión de JDK que se requiera (elegir entre versión 6 y 7)
   java
   alternatives --config java

   There are 5 programs which provide 'java'.

```
  Selection    Command
-----------------------------------------------
*  1           /usr/java/jdk1.6.0_24/jre/bin/java
   2           /usr/lib/jvm/jre-1.5.0-gcj/bin/java
 + 3           /usr/java/jdk1.6.0_26/jre/bin/java
   4           /usr/lib/jvm/jre-1.6.0-openjdk/bin/java
   5           /usr/java/jdk1.7.0_25/jre/bin/java
```

   Enter to keep the current selection[+], or type selection number: 5

   javaws
   alternatives --config javaws
   There are 3 programs which provide 'javaws'.

```
  Selection    Command
-----------------------------------------------
*  1           /usr/java/jdk1.6.0_24/jre/bin/javaws
 + 2           /usr/java/jdk1.6.0_26/jre/bin/javaws
   3           /usr/java/jdk1.7.0_25/jre/bin/javaws
```

   Enter to keep the current selection[+], or type selection number: 3

   javac
   alternatives --config javac

There are 3 programs which provide 'javac'.

```
 Selection    Command
-----------------------------------------------
*  1         /usr/java/jdk1.6.0_24/bin/javac
 + 2         /usr/java/jdk1.6.0_26/bin/javac
   3         /usr/java/jdk1.7.0_25/bin/javac
```

Enter to keep the current selection[+], or type selection number: 3

7. Como proceso de post implementación, se añade la variable de ambiente JAVA_HOME al archivo /etc/profiles o al archivo $HOME/bash_profile
   ## export JAVA_HOME JDK ##
   export JAVA_HOME="/usr/java/jdk1.7.0_25"

Instalación WL en RHEL

1. Descargar el instalador de WebLogic
2. Correr el siguiente comando para comenzar la instalación.
   ./wls1033_linux32.bin
   Esto inicia la instalación en modo gráfico
3. Seleccionar el directorio del home del middleware (Figura 9)

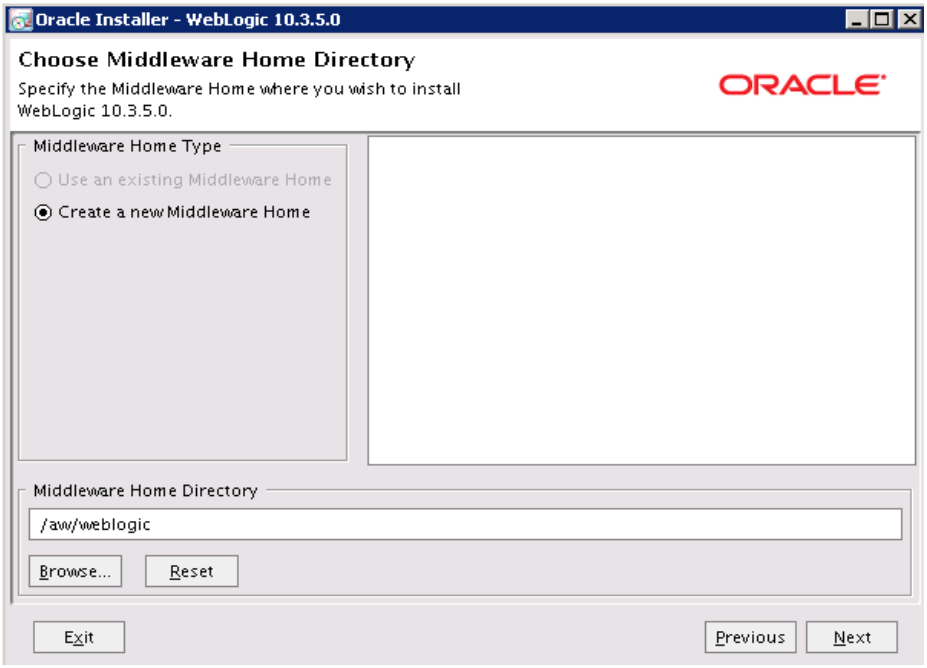

Figura 9. Pantalla de selección del home del middleware

4. Seleccionar el tipo de instalación. Se eligió el modo Custom (Figura 10)

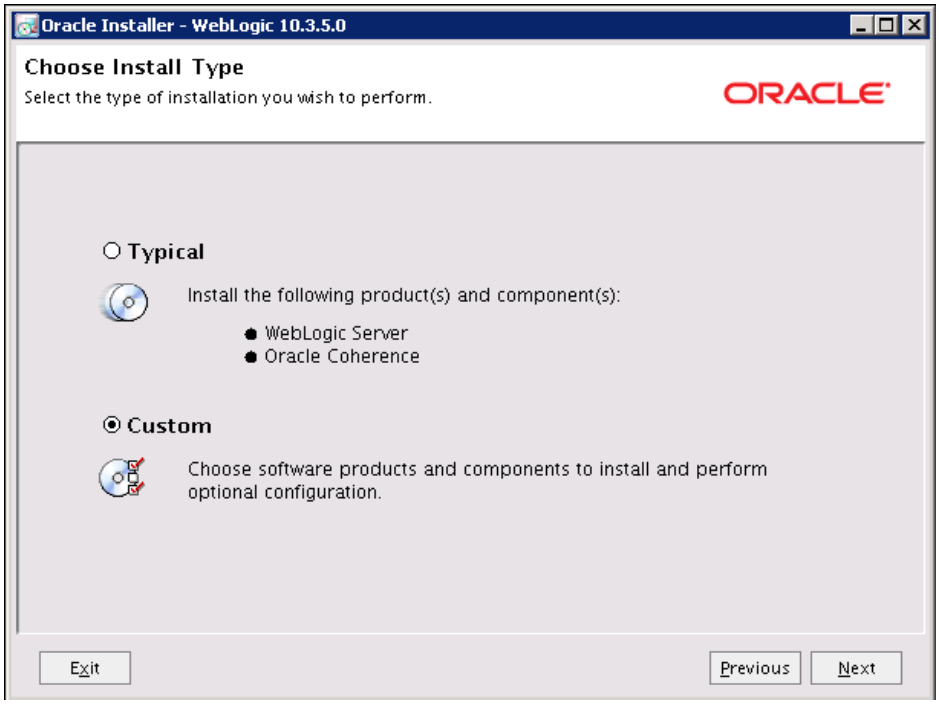

Figura 10. Pantalla de tipo de instalación WebLogic

5. Elegir los componentes a instalar (Figura 11)

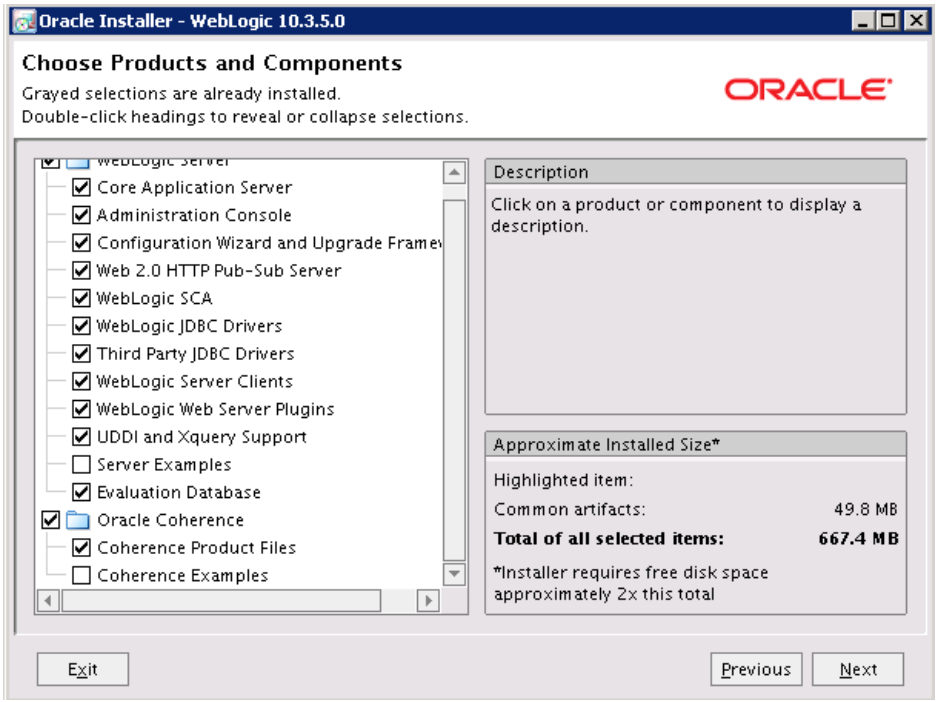

Figura 11. Pantalla de componentes a instalar en WebLogic

6. Seleccionar la versión de JDK que utilizará Weblogic (Figura 12)

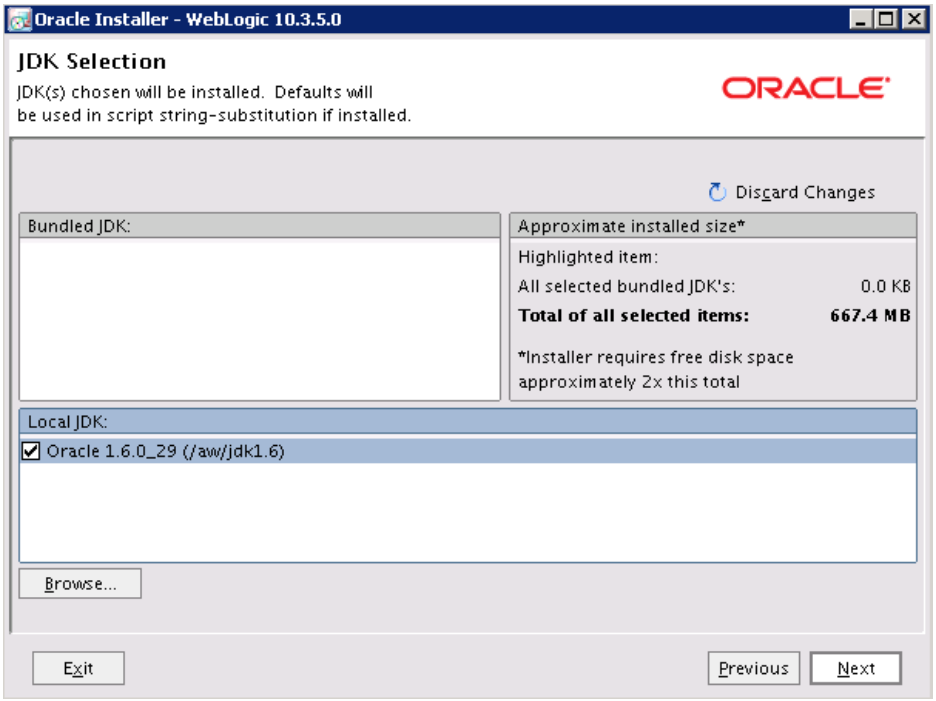

Figura 12. Pantalla de selección de versión de Java

7. Dejar el path default de servidor Weblogic y de Oracle Coherence (Figura 13)

Figura 13. Pantalla de selección de paths para WebLogic

8. Confirmar la instalación y esperar mensaje de confirmación (Figura 14)

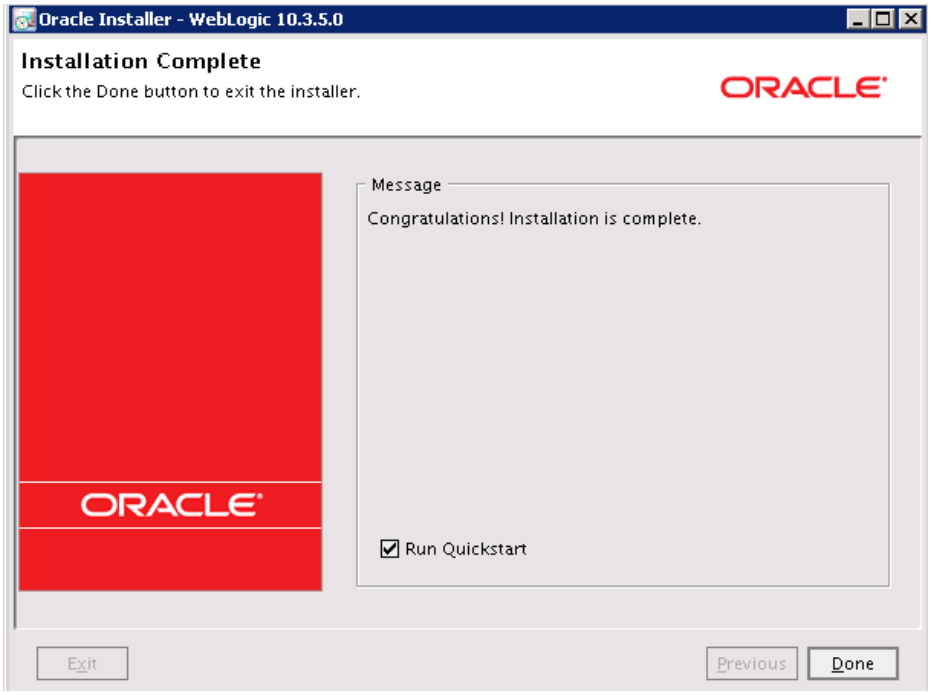

Figura 14. Pantalla de instalación exitosa de WebLogic

Como primer paso, se debería crear el dominio de Weblogic. Se especifican las rutas del home de BEA, el home de Weblogic y el home del nuevo dominio; al detectar que ese dominio no existe, se debe especificar el nombre del mismo y se completa su creación. Una vez creado el dominio, ahora se tendría que establecer los parámetros de la consola de administración para dicho dominio (nombre, puerto TCP a utilizar, nombre de host, user y password).

Terminado el dominio, se debe instalar la configuración de Weblogic; además de los parámetros establecidos en la primera parte de la instalación, se especifica el puerto SSL a utilizar, el número de servidores Weblogic, el hostname de las máquinas en cluster, el puerto del servidor cluster, entre otras. Además, se debe asociar la base de datos creada en el primer paso de la instalación.

Un dominio de Weblogic se compone de un servidor de administración y al menos un servidor manejado. El servidor de administración es responsable de proveer configuraciones para todos los servidores de un dominio específico, así como del monitoreo de los servidores manejados (Figura 15). Los servidores manejados son responsables de ejecutar la lógica de negocios; esto quiere decir que en estos servidores es donde se realiza el despliegue de las aplicaciones entregadas por los desarrolladores.

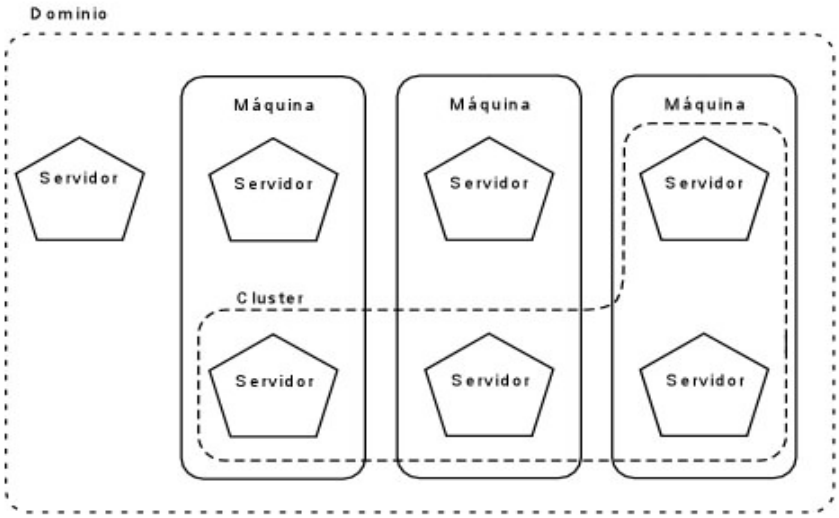

Figura 15: Arquitectura estándar de dominio de WebLogic en cluster

La alternativa para crear un dominio en Weblogic, sin usar el instalador provisto por los desarrolladores, es usar el modo gráfico:

En el path <MIDDLE_HOME>/wlserver_10.3/common/bin, correr el siguiente script: *./config.sh*

1. Seleccionar Crear Nuevo dominio (Figura 16)

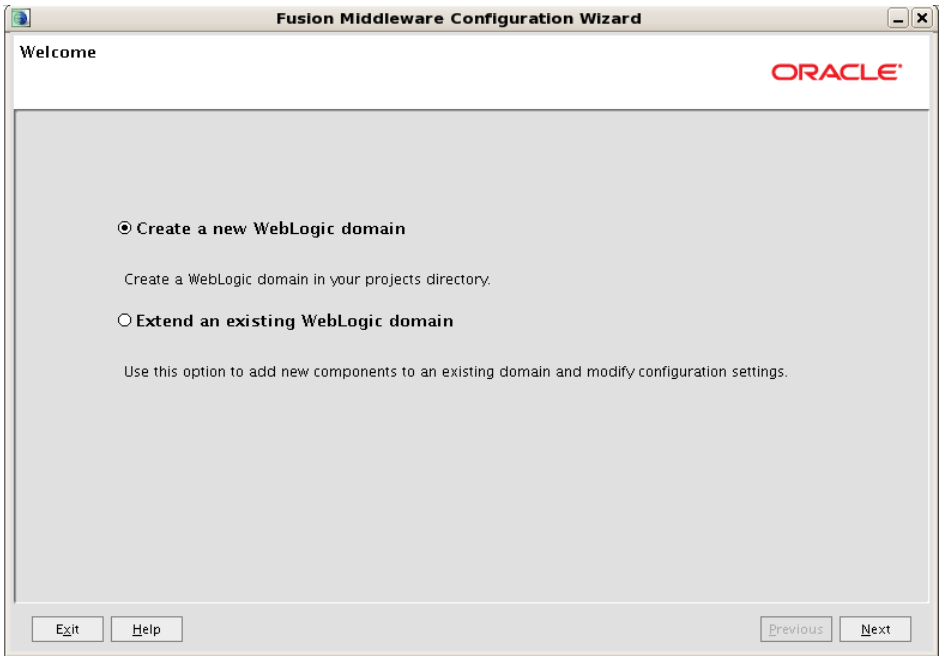

Figura 16. Pantalla de creación de dominio en WebLogic

2. Seleccionar fuente del dominio. Ya está seleccionado por default (Figura 17)

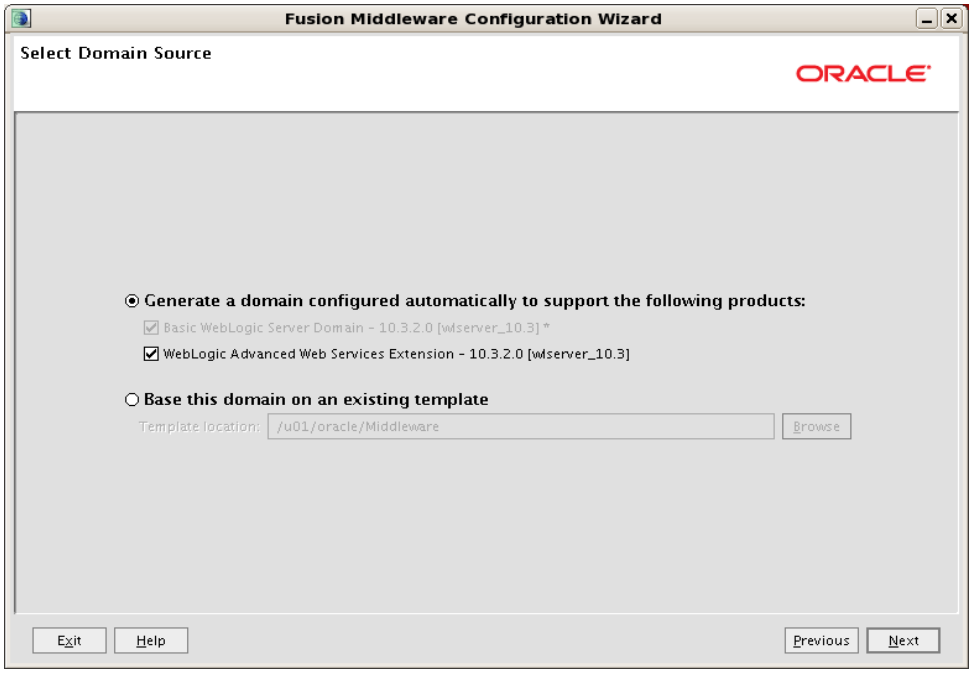

Figura 17. Pantalla de selección de template de dominio en WebLogic

3. Seleccionar nombre del dominio y locación (Figura 18)

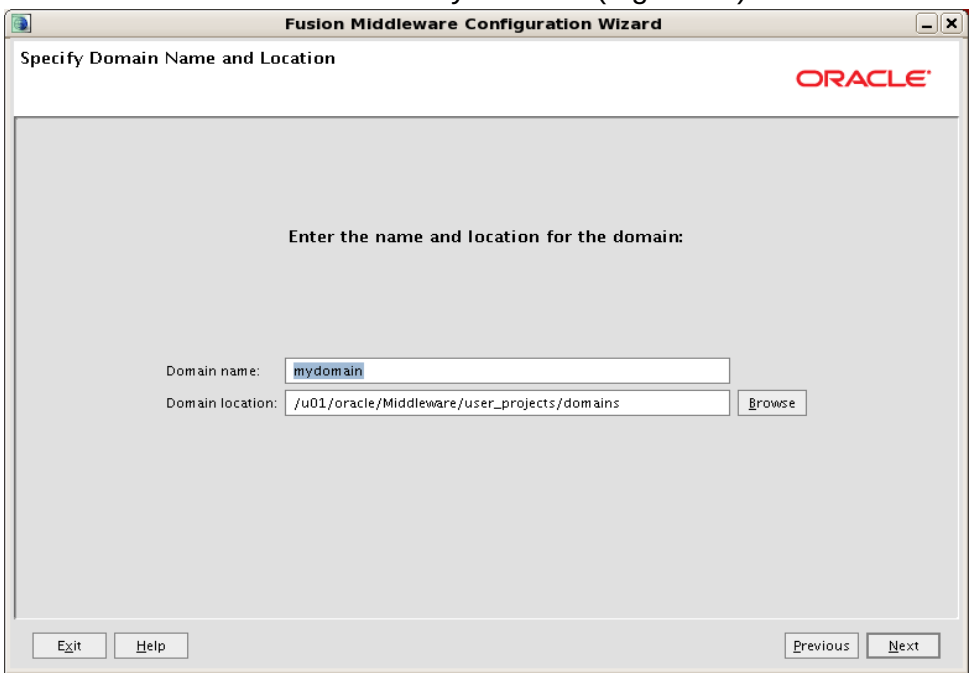

Figura 18. Pantalla de selección de nombre y path del dominio de WebLogic

4. Configurar usuario de administrador y su password correspondiente (Figura 19)

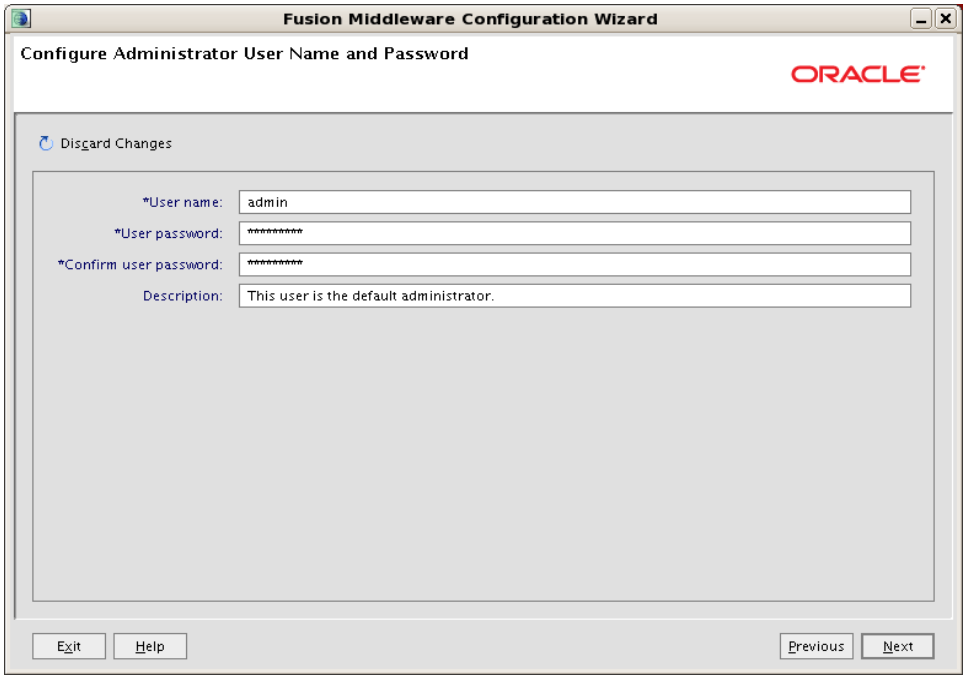

Figura 19. Pantalla de user y password para dominio de WebLogic

5. Seleccionar modo producción. El modo development o desarrollo se recomienda sólo para uso de los desarrolladores (Figura 20)

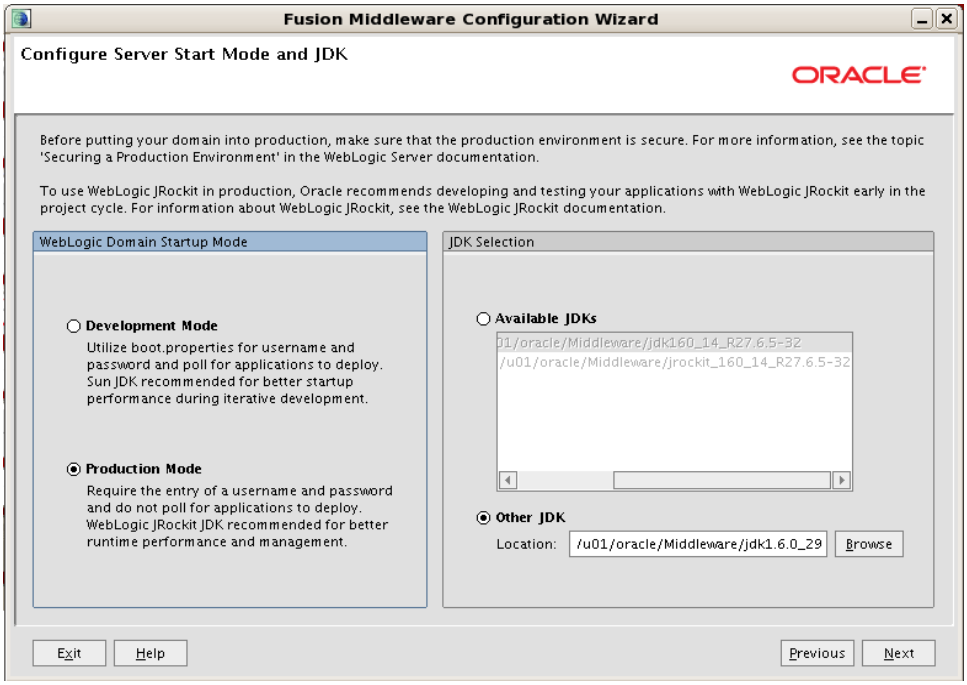

Figura 20. Pantalla de modo de inicio de servidor para dominio de WebLogic

6. Seleccionar configuraciones opcionales. Se selecciona las opciones de Servidor Administrativos y Servidores Manejados, Clusters y Máquinas (Figura 21)

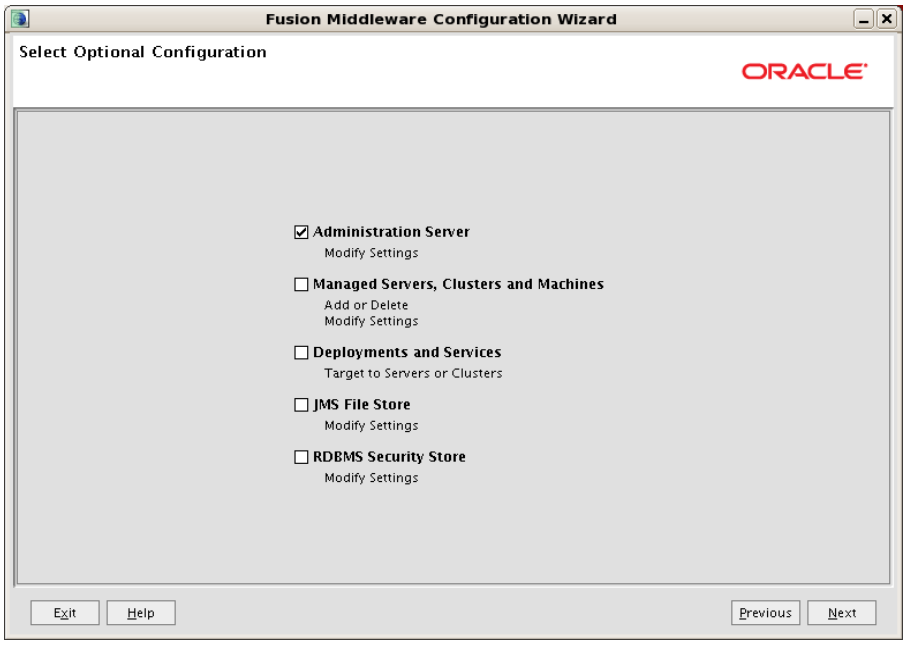

Figura 21. Pantalla de selección de configuraciones adicionales en WebLogic

7. Configurar el servidor de administración (nombre, puerto y puerto SSL)

8. Configurar el servidor manejado (nombre, puerto y puerto SSL. Figura 22)

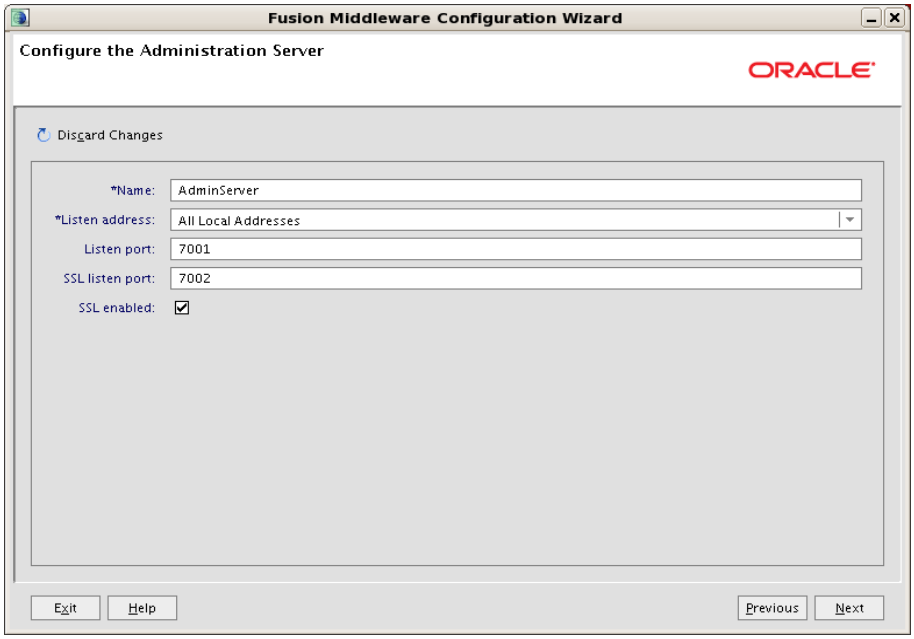

Figura 22. Pantalla de configuración de servidor de administración

9. Configurar cluster y máquinas, si se requiere

10. Confirmar los datos y finalizar el proceso (Figura 23)

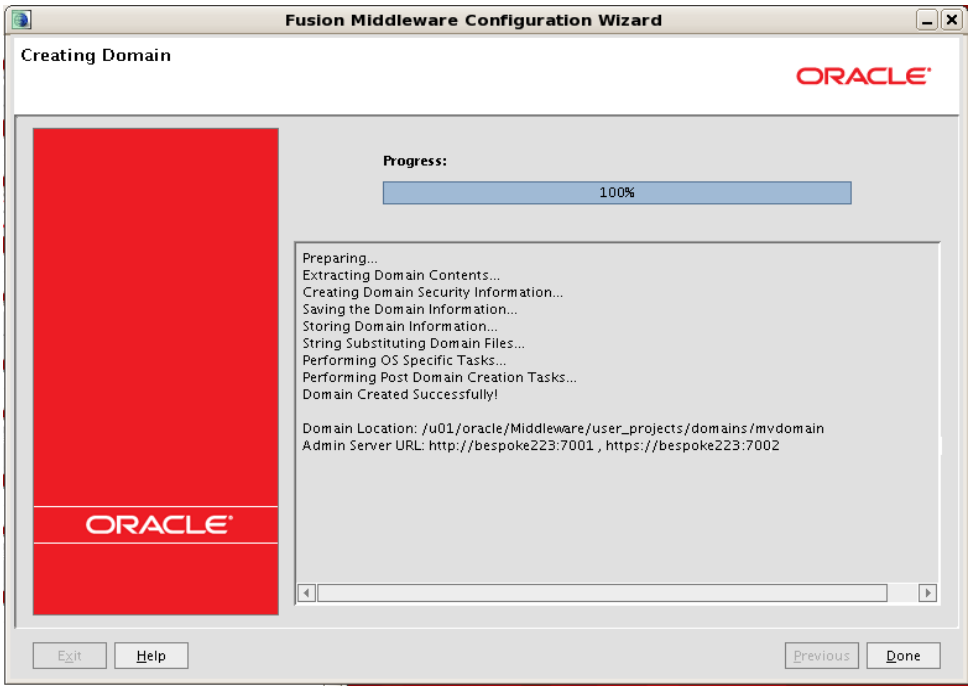

Figura 23. Pantalla de creación exitosa de dominio de WebLogic

Una vez terminada la configuración del servidor de aplicaciones, se iniciaría la instalación de la aplicación como tal. En este proceso se deben tener en cuenta los valores de los puertos, hostnames, variables de entorno y base de datos se deben utilizar. Finalmente, se copia el archivo con extensión .EAR (la aplicación) en la ruta correspondiente para que se realice el despliegue en el servidor manejado.

EAR (*Enterprise Archive*) es un formato de archivo utilizado por Java EE para empaquetar uno o más módulos en un solo archivo para que el despliegue de varios módulos dentro de un servidor de aplicaciones suceda simultánea y coherentemente. También contiene archivos de configuración XML que describen como desplegar los módulos. También existen archivos con extensiones .WAR (aplicaciones web), los cuales no incluyen la lógica de negocios provistas por los EJB's; así como archivos JAR (*Java Archive*), que comúnmente son librerías. Otros componentes de un .EAR son servidores JMS para el manejo de cola de mensajes; drivers JDBC (*Java Database Connectivity*) para manejar conexiones a bases de datos; e interfaces de tipo JNDI (*Java Naming and Directory Interface*).

Para verificar que se instaló correctamente, se deben ejecutar los scripts de inicio y apagado correspondientes y verificar que los procesos se levanten y se finalicen,

respectivamente. Con la aplicación iniciado, se deben realizar pruebas de sanidad correspondiente para verificar de manera mínima que el sistema esté funcionando correctamente.

Una vez que la instalación ha concluido y que las pruebas de sanidad han sido completadas satisfactoriamente, se inicia el periodo de pruebas. En primera instancia, se llevan a cabo las pruebas de conectividad.

Pruebas de conectividad

Una VPN (*Virtual Private Network*) es una estructura de red corporativa implantada sobre una red de recursos de carácter público, pero que utiliza el mismo sistema de gestión y las mismas políticas de acceso que se usan en las redes privadas; al fin y al cabo no es más que la creación en una red pública de un entorno de carácter confidencial y privado que permitirá trabajar al usuario como si estuviera en su misma red local. Dado que el sistema de pagos móviles tiene que comunicarse con los sistemas de los clientes, se crearon reglas de conectividad las cuales describen cuáles serán las direcciones IP y puertos visibles para cliente y viceversa. Se realizan las pruebas de conexión con cada de unas las VPN's establecidas para garantizar la comunicación entre cada ambiente de pruebas.

En este punto se utiliza el principio de NAT (Network address translation). La conversión de direcciones de red o NAT se desarrolló para resolver la falta de direcciones IP con el protocolo IPv4. El principio de NAT consiste en utilizar una conexión de pasarela a Internet, que tenga al menos una interfaz de red conectada a la red interna y al menos una interfaz de red conectada a Internet (con una dirección IP enrutable) para poder conectar todos los equipos a la red.

Por lo tanto, se configura cada equipo en la red que necesite acceso a Internet para que utilice una pasarela de NAT (al especificar la dirección IP de la pasarela en el campo gateway con sus parámetros TCP/IP). Cuando un equipo de red envía una solicitud a Internet, la pasarela hace la solicitud en su lugar, recibe la respuesta y la envía al equipo que hizo la solicitud. Debido a que la pasarela oculta completamente las direcciones internas en la red, el mecanismo de conversión de direcciones de red brinda una función segura. De hecho, para un observador externo de la red, todas las solicitudes parecen provenir de la dirección IP de pasarela.

Telnet es un protocolo que sirve para emular una terminal remota, lo que significa que se puede utilizar para ejecutar comandos introducidos con un teclado en un equipo remoto. La herramienta Telnet está implementada por el protocolo Telnet. Esto significa que traduce las especificaciones del protocolo al lenguaje de programación a fin de crear un programa que pueda emular una terminal. Telnet opera en un entorno de cliente/servidor, lo que implica que el equipo remoto se configura como servidor, por lo que espera que el otro equipo le solicite un servicio. Por lo tanto, dado que este

equipo remoto envía datos que se deben mostrar, el usuario siente que está trabajando directamente en una máquina remota.

El protocolo Telnet se aplica en una conexión TCP para enviar datos en formato ASCII codificados en 8 bits, entre los cuales se encuentran secuencias de verificación Telnet. Por lo tanto, brinda un sistema de comunicación orientado bidireccional (semidúplex) codificado en 8 bits y fácil de implementar.

El comando para iniciar una sesión Telnet generalmente es:

telnet nombre_del_servidor

nombre_del_servidor representa el nombre o la dirección IP del equipo remoto al que se quiere conectar el usuario. También puede usar su dirección IP, por ejemplo:

telnet 125.64.124.77

Por último, también puede especificar el puerto que desea usar introduciendo el número de puerto después de la dirección IP o el nombre del servidor:

telnet 125.64.124.77 80

Para que las pruebas de conectividad sean exitosas, se deben configurar las reglas de firewall necesarias en ambas direcciones para asegurar que el tráfico de red entre un servidor y otro. Un cortafuegos o firewall es un sistema de defensa basado en el hecho de que todo el tráfico de entrada o salida a la red debe pasar obligatoriamente por un sistema de seguridad capaz de autorizar, denegar y tomar nota de todo aquello que ocurre, de acuerdo con una política de control de acceso entre redes. Controla tanto la comunicación desde el exterior como el tráfico generado desde la propia máquina o red interna. Actúa a base de normas que establece el administrador de seguridad o, en su defecto, el administrador de red o el usuario final. Dichas reglas definen las acciones correspondientes a llevar a cabo cuando se recibe un paquete que cumpla unas determinadas características.

Por cuestiones de seguridad, se sugiere configurar un proxy reverso para que las peticiones HTTP del cliente lleguen al servidor por los puertos conocidos (80 para HTTP y 443 para HTTPS). El servidor de proxy inverso es utilizado como un intermediario por los usuarios de Internet que desean acceder a un sitio web interno al enviar sus solicitudes indirectamente. Con un proxy inverso, el servidor web está protegido de ataques externos directos, lo cual fortalece la red interna (Figura 24).

Además, la función caché de un proxy inverso puede disminuir la carga de trabajo del servidor asignado, razón por la cual se lo denomina en ocasiones acelerador de servidor. Finalmente, con algoritmos perfeccionados, el proxy inverso puede distribuir

la carga de trabajo mediante la redirección de las solicitudes a otros servidores similares. Este proceso se denomina equilibrio de carga.

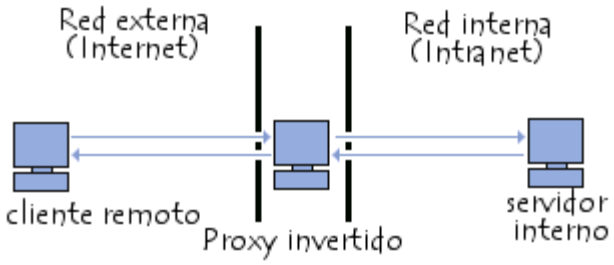

Figura 24. Estructura básica de proxy invertido

Por otra parte, al utilizar un servidor proxy, las conexiones pueden rastrearse al crear registros de actividad (logs) para guardar sistemáticamente las peticiones de los usuarios cuando solicitan conexiones a Internet. Gracias a esto, las conexiones de Internet pueden filtrarse al analizar tanto las solicitudes del cliente como las respuestas del servidor. El filtrado que se realiza comparando la solicitud del cliente con una lista de solicitudes autorizadas se denomina lista blanca; y el filtrado que se realiza con una lista de sitios prohibidos se denomina lista negra. El análisis de las respuestas del servidor que cumplen con una lista de criterios (como palabras clave) se denomina filtrado de contenido.

Un host virtual es una forma de poder utilizar varios dominios basados en una misma IP. Esto facilita el trabajo en un servidor local ya que podemos crear un dominio para cada aplicación que estemos desarrollando.

Ejemplo de 2 Hosts virtuales (HTTP y HTTPS)

<***VirtualHost *:80***>

ServerName estoesunaprueba.com

ProxyPreserveHost on *(Esta línea en ON hace que cuando la consulta llega al Proxy, éste mantenga el nombre de la consulta y no el definido en ProxyPass. En este caso mantiene el nombre estoesunaprueba.com en lugar de mostrar la IP 192.168.1.110 junto a su puerto)*
ProxyPass / http://192.168.1.110:8885/prueba *(la primera / indica cómo el home de acceso va a tomar lo que se defina después, que en este caso significa que al acceder con el navegador, entre directamente al directorio prueba que tengo configurado en el Servidor Web original)*
ProxyPassReverse / http://192.168.1.110:8885/prueba *(esta opción permite ajustar la URL en los headers de la consulta HTTP. Es importante ponerla junto a ProxyPass)*
</VirtualHost>

<VirtualHost *:443>

ProxyPreserveHost On
ProxyPass / http://192.168.1.110:8885/prueba
ProxyPassReverse http://192.168.1.110:8885/prueba

ServerName estoesunaprueba.com

SSLProxyEngine On *(Activa al proxy para que funcione con ssl,)*
SSLEngine on *(Módulo de ssl propiamente dicho activo)*
SSLCertificateFile /etc/apache2/ssl/cert.crt *(Certificado para ssl)*
SSLCertificateKeyFile /etc/apache2/ssl/cert.key *(Key para ssl)*

</VirtualHost>

Una vez que todas las VPN's han sido probadas de manera exitosa, se dan por concluidas las pruebas de conectividad.

Pruebas DEV2DEV

La primera etapa de pruebas se le considera pruebas DEV2DEV. En este tipo de pruebas se deja de lado la funcionalidad del sistema y simplemente se verifica que los inputs y outputs del sistema cumplan con la especificación; dado que en esta etapa de pruebas el sistema no ha sido refinado de manera completa, la tarea de probar flujos completos no es recomendada y sólo se determina que los parámetros de entrada y salida cumplan con lo que ha sido especificado en el API (*Application Programming Interface*).

Una API es un conjunto de funciones que permite al programador acceder a servicios de una aplicación a través del uso de un lenguaje de programación. Una API ofrece al programador un cierto nivel de abstracción que enmascara la complejidad de acceso a un sistema o aplicación, proponiéndole un conjunto de funciones de las cuales sólo se conocen los parámetros y los valores devueltos. Gracias a las API, un desarrollador no necesita preocuparse de cómo funciona una aplicación remota ni de la forma en que las funciones fueron implementadas, para poder utilizarla en un programa. Una API puede estar disponible para un lenguaje específico o para diversos lenguajes de programación.

Por lo mismo, se deja de lado a los usuarios finales y las pruebas se realizan por lo general entre la gente técnica de ambos lados (clientes y proveedor). Como toda fase de pruebas, el objetivo principal es encontrar la mayor cantidad de bugs (errores o anomalías) posibles, para informar a los desarrolladores y que hagan los arreglos necesarios. Dada que la cantidad de errores es considerable en esta etapa, el involucramiento de los desarrolladores es mayor en esta etapa.

Para recabar la información necesaria del funcionamiento y los errores del sistema, los desarrolladores utilizan la biblioteca log4j para generar archivos de logs. Log4j es una API escrita en Java bajo la licencia Apache Software que también es portable para C, C#, Perl, Python, Ruby, entre otros. Es altamente configurable a través de archivos de configuración. Muestra el proceso de logging en término de niveles de prioridades y ofrece mecanismo de logging para una gran variedad de destinos como base de datos, archivos, consola, UNIX Syslog, etc.

Log4j maneja distintos niveles de prioridad:

- FATAL: se utiliza para mensajes críticos del sistema, generalmente después de guardar el mensaje el programa abortará.
- ERROR: se utiliza en mensajes de error de la aplicación que se desea guardar, estos eventos afectan al programa pero lo dejan seguir funcionando.
- WARN: se utiliza para mensajes de alerta sobre eventos que se desea mantener constancia, pero que no afectan al correcto funcionamiento del programa.
- INFO: designa mensajes informativos.
- DEBUG: se utiliza para escribir mensajes de depuración. Este nivel no debe estar activado cuando la aplicación se encuentre en producción.
- TRACE: se utiliza para mostrar mensajes con un mayor nivel de detalle que debug.

Para pruebas DEV2DEV, el nivel de log que se maneja es menos restrictivo que en otros ambientes (en ciertos casos se utiliza DEBUG). Cuando el sistema pasa a producción, el nivel de log se reduce considerablemente, pues la cantidad de información detectada es demasiada y esto generaría archivos demasiado grandes que pueden afectar la operación.

Los protocolos de comunicación que se recomiendan emplear entre los clientes y la aplicación son Webservices e ISO 8583.

El término Web Services o servicios Web describe una forma estandarizada de integrar aplicaciones WEB mediante el uso de XML (*Extensible Markup Language*), SOAP (*Simple Object Access Protocol*), WSDL (*Web Services Description Language*) y UDDI (*Universal Description, Discovery and Integration)* sobre los protocolos de la Internet. XML es usado para describir los datos, SOAP se ocupa para la transferencia de los datos, WSDL se emplea para describir los servicios disponibles y UDDI se ocupa para conocer cuáles son los servicios disponibles (Figura 25). Uno de los usos principales es permitir la comunicación entre las empresas y sus clientes. Los servicios Web permiten a las organizaciones intercambiar datos sin necesidad de conocer los detalles de sus respectivos Sistemas de Información.

Los servicios Web permiten a distintas aplicaciones, de diferentes orígenes, comunicarse entre ellos sin necesidad de escribir programas costosos, esto porque la comunicación se hace con XML. Los servicios Web no están ligados a ningún sistema operativo o lenguaje de programación. Por ejemplo, un programa escrito en Java puede conversar con otro escrito en Pearl; aplicaciones Windows puede conversar con aplicaciones Unix. Por otra parte los servicios Web no necesitan usar navegadores (Explorer) ni el lenguaje de especificación HTML (*HyperText Markup Language*).

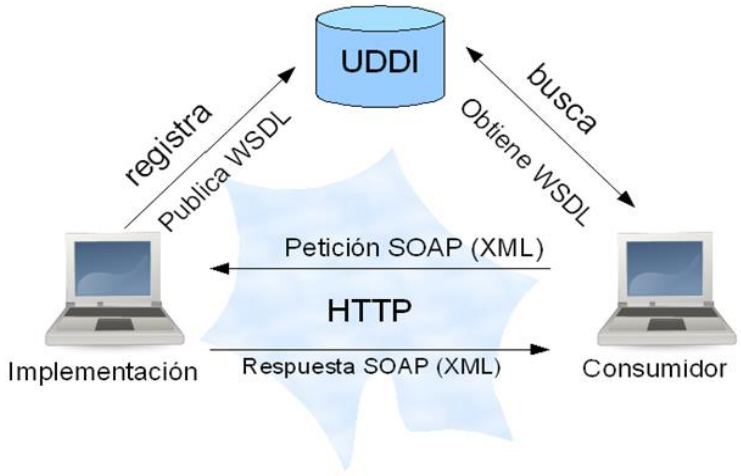

Figura 25. Arquitectura básica de los Web Services

ISO 8583 provee un framework para crear protocolos para el intercambio de mensajes de transacciones financieras. Típicamente, éstos son mensajes que involucran transacciones de algunas tarjetas de algún tipo, como de débito o de crédito. ISO8583 se considera como un metaprotocolo que provee un conjunto de reglas para la definición de protocolos de transacciones financieras.

Debido a esto, existen una variedad de protocolos ISO8583, siendo de los famosos el protocolo BASE I de Visa para autorización de transacciones. Hay protocolos propietarios de terminales POS usados para comunicarse con sus hosts, y registros de otros estándares definidos por fabricantes de equipos, vendedores de software, bancos o asociaciones de estandarización. Es muy común que en los pagos con tarjetas, sea un ATM (cajero automático) o una terminal POS (punto de venta), la transacción sea transportada por un protocolo de la familia ISO8583.

A diferencia de RFC (*Request For Comments*) o Ecma International, el estándar ISO 8583 no está disponible de manera gratuita y no hay algún lugar en Internet donde pueda ser descargado. Esto aplica para la mayoría de los estándares publicados por ISO. El estándar puede ser adquirido en la página oficial de ISO y cuesta alrededor de 150 dólares.

Pruebas integrales

Finalizada esta etapa, se debe iniciar la etapa de pruebas integrales. Una vez que el sistema se encuentra más estable, las siguientes pruebas ya involucran datos de prueba realistas y se comienzan a probar flujos completos por parte del cliente (particularmente, el área de negocio); también se planea completar la integración del core o núcleo con el resto de los componentes customizables. Igualmente, se encuentran y documentan los errores encontrados, tanto en el núcleo como en los customs, se realiza el informe y se entrega al desarrollador para que lleve a cabo las correcciones necesarias; mencionar que los bugs tienen cierta prioridad y criticidad, y dependiendo de estos factores, se determina que errores se corregirán primero.
Para la instalación de los módulos customs, se requeriría instalar otra instancia de Weblogic en donde se realice el despliegue de estos nuevos componentes. Resaltar que una o varias instancias del servidor de aplicaciones pueden ser alojadas en una misma máquina servidor.

Los módulos customizables para el sistema de pagos móviles cubrirían las siguientes necesidades:

- Módulo de web self service (portal en línea): Permite a los clientes actualizar información, solicitar información o solicitar soporte cuando lo necesite entrando a un sitio web
- Módulo de atención telefónica: Software utilizado por la compañía que proporciona la tecnología de IVR para soportar el servicio de banca telefónica. Permite que el cliente pueda hacer llamadas para solicitar información o realizar operaciones.
- Módulo de estados de cuenta: Generará los estados de cuenta de todos los usuarios de cada banco.
- Módulo de sucursal: permite al operador de sucursal realizar registros, altas de tarjetas, edición de datos, entre otras operaciones.
- Módulo de gestión core-custom-proveedor telefonía celular: se encarga de gestionar la comunicación entre el core con los servidores de USSD de la compañía telefónica para lograr la interacción con el usuario de telefonía celular. También se encarga de entregar las notificaciones SMS a los clientes.
- Módulo antifraude: Permite el monitoreo de transacciones y altas de clientes con el propósito de evitar comportamiento fraudulento en la plataforma a través de validación de listas blancas y negras

La comunicación entre estos componentes y el núcleo se puede establecer a través de varios protocolos como son EJB (*Enterprise Java Beans*), servicios Web y SP (*Store Procedures*).

Los EJB proporcionan un modelo de componentes distribuido estándar del lado del servidor. El objetivo de los EJB es dotar al programador de un modelo que le permita

abstraerse de los problemas generales de una aplicación empresarial (concurrencia, transacciones, persistencia, seguridad, etc.) para centrarse en el desarrollo de la lógica de negocio en sí. El funcionamiento de los componentes EJB se basa fundamentalmente en el trabajo del contenedor EJB. El contenedor EJB es un programa Java que corre en el servidor y que contiene todas las clases y objetos necesarios para el correcto funcionamiento de los enterprise beans (Figura 26).

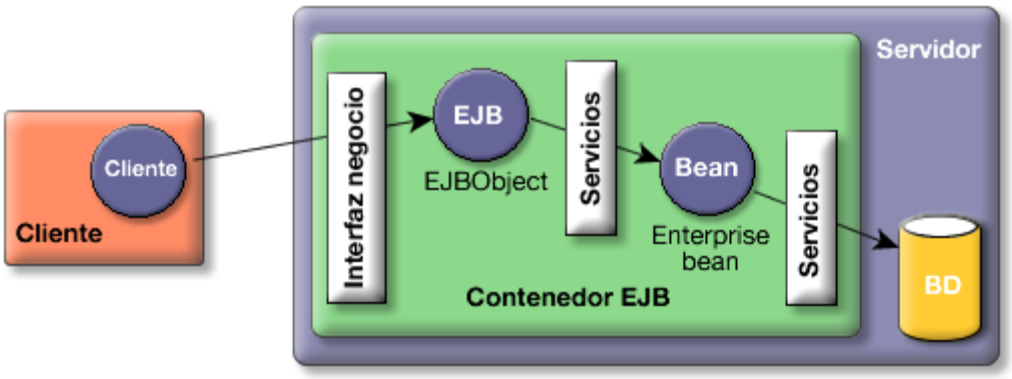

Figura 26. Estructura básica EJB

La tecnología EJB define tres tipos de beans: beans de sesión, beans de entidad y beans dirigidos por mensajes. Los beans de entidad representan un objeto concreto que tiene existencia en alguna base de datos de la empresa; una instancia de un bean de entidad representa una fila en una tabla de la base de datos. Los beans dirigidos por mensajes pueden escuchar mensajes de un servicio de mensajes JMS (*Java Message Service*); los clientes de estos beans nunca los llaman directamente, sino que es necesario enviar un mensaje JMS para comunicarse con ellos. Un bean de sesión representa un proceso o una acción de negocio.

Los SP o procedimientos almacenados son subrutinas disponibles para que las aplicaciones accedan a una base de datos relacional. Los procedimientos almacenados están contenidos en el diccionario de datos de la base de datos, además de ser enunciados y pre-compilados (Figura 27). Los usos típicos para los SP incluyen validación de datos o mecanismos para control de acceso; además de que se usan para consolidar y centralizar lógica que fue implementada originalmente en aplicaciones. Procesamiento excesivo o complejo que requiera la ejecución de muchas instrucciones SQL es movido a los SP, así todas las aplicaciones pueden llamar a esos procedimientos.

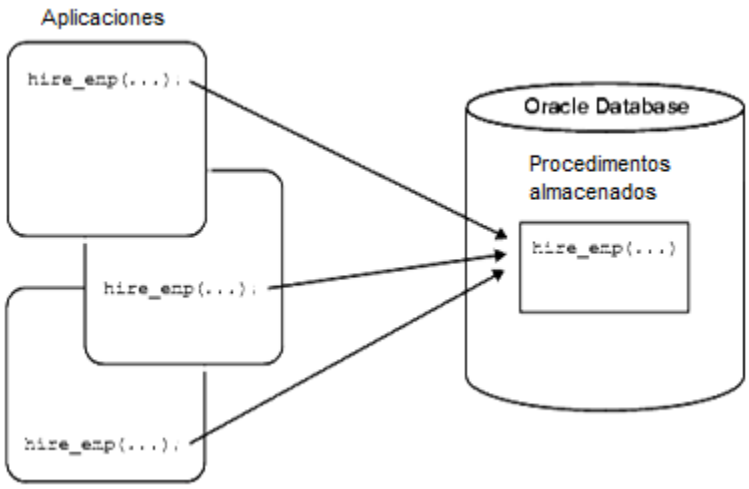

Figura 27. Utilización básica de Store Procedures

Pruebas UAT

Concluida la integración de estos componentes y con el visto bueno del cliente durante la etapa de integración (significando esto que se han resuelto la suficiente cantidad de bugs bloqueantes), la nueva versión del sistema pasaría al ambiente de UAT.

En las pruebas de UAT, cada banco tiene varias áreas dedicadas específicamente a probar cada módulo del sistema (áreas de negocio, área de atención telefónica, área de estados de cuenta, área de reportes, etc.); esto significa que las pruebas se hacen más exhaustivas y la posibilidad de encontrar errores se incrementan, puesto que se encuentran escenarios de pruebas no previstos durante etapas anteriores.

Tanto Gemalto como los bancos presentan su plan de pruebas con el propósito de contemplar la mayor cantidad de escenarios posibles por probar; tanto escenarios de prueba exitosos (conocidos como happy paths) como escenarios con el propósito de ser fallidos. Puesto que la visibilidad se hace mayor en este ciclo de pruebas, se hace crucial resolver los bugs más críticos en el menor tiempo posible.

El ciclo de vida de un bug o incidencia es el siguiente (Figura 28):

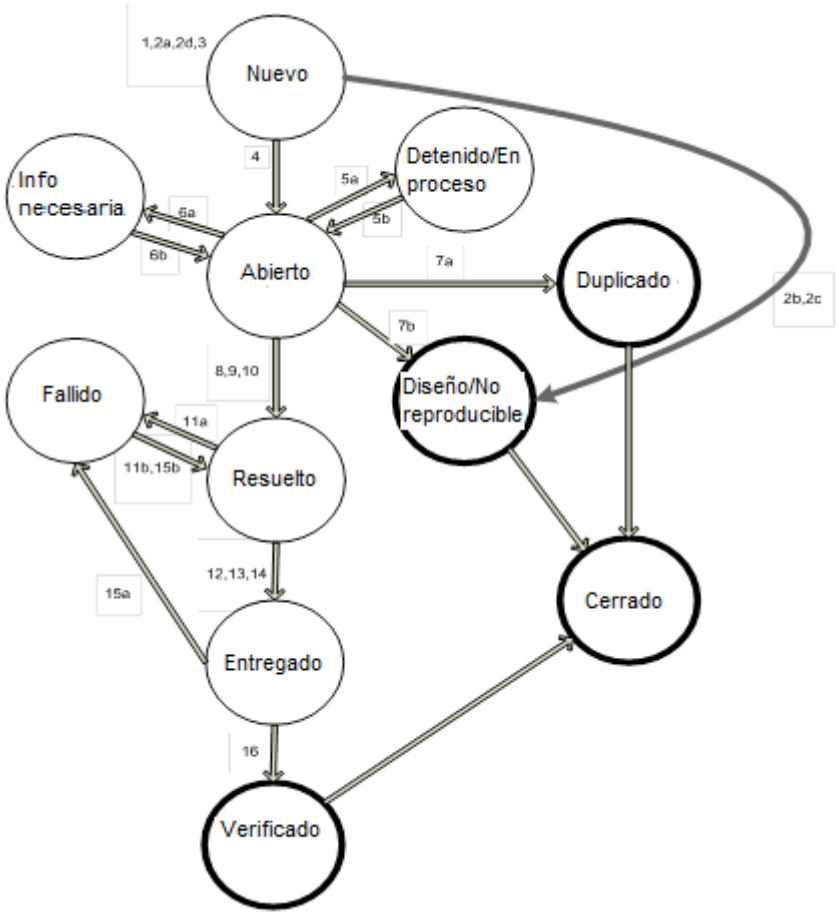

Figura 28. Ciclo de vida de defectos en sistema

Cuando el cliente reporta una incidencia, se realiza el análisis correspondiente y se determina si en efecto es un error. En caso de que el comportamiento sea de acuerdo al diseño o el error no sea reproducible, se notifica al cliente para cerrar el caso; esto también ocurre cuando el cliente reporta un caso duplicado.

Se dan casos en donde los desarrolladores requieren más evidencia del error o se necesita clarificación del comportamiento esperado; si se presentan dichos casos el bug pasa a estado de Información requerido/en espera hasta que nueva información se les sea proporcionada.

Una vez que se tiene claro el escenario de falla, se procede a su corrección. Una vez que se tenga la solución necesaria, debe pasar por varios filtros; primeramente revisión de código, luego pruebas en un ambiente de pruebas internos y finalmente pruebas en ambientes de prueba con el cliente. Si en alguna de estas validaciones se encuentra un error, la solución de la incidencia se considera fallida y se inicia el proceso nuevamente.

Finalmente, si la solución del error cumple con la expectativa del cliente se da por cerrado el caso, confirmando con el desarrollador que su arreglo ha sido exitoso.

Para la instalación de las actualizaciones o parches, se debe establecer una ventana de mantenimiento con el consentimiento de los clientes, determinando los componentes a ser actualizados y el tiempo estimado en el cual la plataforma no estará disponible. Esto es obligatorio para el ambiente de UAT, pues es necesario comprobar que dichos mantenimientos no afectarán las pruebas que tengan planeadas el cliente durante el transcurso del día; para pruebas DEV2DEV o Integrales, no es requerido la autorización de ventanas de mantenimiento.

Durante la etapa final de las pruebas UAT es responsabilidad de Gemalto proveer los documentos de ATP (Plan de aceptación de pruebas) a los clientes. Se genera un ATP por cada requerimiento que fue implementado y probado; cada documento incluye los casos de pruebas que fueron probados, los datos de prueba utilizados, los responsables de cada parte (tanto proveedor como cliente) de la ejecución de las pruebas y el estatus de las casos ejecutados. Se les entrega a los clientes y ellos deben regresar dichos documentos firmados dando el consentimiento de que la información contenida se apega a lo que se ejecutó.

Pruebas de estrés y alta disponibilidad

También se contemplan otro tipo de pruebas, como son pruebas de estrés y pruebas de alta disponibilidad.

Las pruebas de estrés son una forma de probar intensamente de manera deliberada para determinar la estabilidad del sistema. Se hacen pruebas de este tipo para identificar “cuellos de botella”, reducir el riesgo de un colapso del sistema, aprovechar los recursos de mejor manera, conocer los límites que soporta el sistema, permitir tomar decisiones sobre configuraciones de hardware, ajustes de software y revisión de la arquitectura. En esta fase se utilizan pruebas no funcionales.

Una de las herramientas que pueden ser utilizadas para estas pruebas es Apache JMeter. JMeter es una aplicación Java open source diseñada para determinar el comportamiento en pruebas de carga y medición de performance. Es usada para probar los recursos tanto estáticos como dinámicos (archivos, servlets, scripts en Perl, objetos Java, bases de datos, servidores FTP (File Transfer Protocol), entre otros). Puede simular carga pesada en un servidor, una red o un objeto para probar su resistencia o analizar el comportamiento en general. Maneja diferentes tipos de servicios como son:

- Web - HTTP, HTTPS
- SOAP
- Database via JDBC
- LDAP

- JMS
- Mail – SMTP (Simple Mail Transfer Protocol), POP3 e IMAP
- Comandos nativos o scripts de shell

La clave de estas pruebas es asegurar que la plataforma soporta determinada cantidad de TPS. Un TPS es una medida de hardware o software que representa el número de transacciones completadas en un segundo por un sistema de información. La medida de TPS se usa para calcular el desempeño de sistemas que manejan transacciones rutinarias y manejo de estadísticas.

La disponibilidad es el grado en que una aplicación o servicio está disponible cuándo y cómo los usuarios esperan. La disponibilidad se mide por la percepción de una aplicación del usuario final. Los usuarios finales experimentan frustración cuando sus datos no están disponibles y ellos no entienden o son capaces de diferenciar los complejos componentes de una solución global. Fiabilidad, valorización, continuas operaciones y detección de errores son características de una solución de alta disponibilidad.

El balanceo de carga es la manera en que las peticiones de Internet son distribuidas sobre una fila de servidores. Existen varios métodos para realizar el balanceo de carga. Desde el simple "Round Robin" (repartiendo todas las peticiones que llegan de Internet entre el número de servidores disponibles para dicho servicio) hasta los equipos que reciben las peticiones, recogen información, en tiempo real, de la capacidad operativa de los equipos y la utilizan para dirigir dichas peticiones individualmente al servidor que se encuentre en mejor disposición de prestar el servicio adecuado. Los balanceadores de carga pueden ser soluciones hardware, tales como ruteadores y switches que incluyen software de balanceo de carga preparado para ello, así como soluciones de software que se instalan en el back end de los servidores.

El balanceo de carga más seguro sólo se puede conseguir considerando el uso real de los servidores, permitiendo que los recursos existentes se empleen al máximo, al conocer cómo están siendo utilizados estos recursos incluso antes de que las peticiones de los clientes lleguen a ellos. El tráfico se dirige proactivamente, cambiando el antiguo concepto existente de balanceo de carga, hacia una solución de optimización del servidor, consiguiendo el mejor resultado posible con la tecnología disponible. Para lograrlo, el balanceador de carga continuamente realiza peticiones de datos de cada servidor en la granja de servidores para monitorizar sus condiciones y direccionar las peticiones de los clientes hacia el servidor que se encuentre más disponible y en mejor estado para responder a dichas peticiones. Los parámetros solicitados, dependen del producto utilizado. Normalmente se emplea la utilización de la CPU del servidor, el uso de memoria y el número de conexiones abiertas.

Cambios de requerimiento

Durante la etapa de pruebas hay ocasiones donde los clientes no están de acuerdo con alguna funcionalidad de algunos flujos del sistema; en estos casos se solicitan cambios de requerimiento, los cuales implican que se escriban nuevos requerimientos o se realicen cambios en los ya existentes. Esto también implica que se necesitan nuevos desarrollos y se afectan en los tiempos que se tenían contemplados inicialmente para el periodo de pruebas, particularmente alargando la etapa de aceptación (UAT); obviamente, esto implica un costo extra para los clientes por lo que se cotiza el esfuerzo de desarrollo, implementación y pruebas para esas nuevas funcionalidades de una manera diferenciada a lo cotizado previamente.

Dependiendo de la criticidad que establezca el cliente para cada cambio particular, se debe determinar si son entregados lo más pronto posible o si pueden ser retrasados para ser implementados en etapas posteriores para no afectar las fechas comprometidas de salida a producción.

Pase a producción

Cuando la versión implementada tiene corregido todos los errores que el cliente considera bloqueantes para su operación, se convoca a una reunión entre los gerentes de ambas partes (tanto bancos como el proveedor) y se determina si se realizará el pase a producción. Es de suma importancia que el equipo de operaciones se involucre en la etapa de pruebas, pues aunque la versión del sistema será la misma que en el sistema de pruebas, las condiciones difieren de gran manera. Puede que las pruebas permitan que salgan a la luz problemas de desempeño o de seguridad que puedan afectar la operación y que deben ser resueltos antes de considerar el paso a producción

Si se toma la determinación de migrar a la nueva versión, el equipo de operaciones comienza con los preparativos para la ventana de mantenimiento. Generalmente estas ventanas se realizan durante la madrugada de los fines de semana, pues es indispensable que la plataforma esté abajo el menor tiempo posible en un horario donde el número de transacciones se reduzca considerablemente. Y a diferencia de las ventanas de mantenimiento que se llevan a cabo en los ambientes de prueba, para producción se lleva a cabo un proceso mucho más cuidadoso y preciso; pues cualquier afectación al momento de la actualización puede traer graves consecuencias a nivel operación, con clientes y transacciones reales.

Por lo general estas ventanas tienen una duración de varias horas, dado que es importante reservar cierto tiempo de holgura por si algún incidente ocurre durante la actualización; si por alguna razón la actualización falla, se debe realizar un proceso de rollback de los cambios para volver a la versión que estaba implementada anteriormente.

Capítulo 4: Resultados y aportaciones

Cuando inicié el trabajo en octubre de 2011, el porcentaje de incidencias resueltas y cerradas en el proceso de pruebas en el banco asignado era del 50 por ciento. A partir de que se documentaron de mejor manera los escenarios de prueba y se recabaron evidencias más claras y precisas de los errores encontrados, el porcentaje subió hasta un 90 por ciento. Esto permitió que el banco diera la autorización para que el sistema recibiera el OK y pasara a la etapa de salida a producción. La mejora en la administración de las pruebas en UAT permitió que liberaciones posteriores de versiones nuevas del sistema fueran certificadas de una manera más eficiente y más apegada a los tiempos para que pasaran a producción.

Al leer y revisar la documentación disponible del sistema, comprendí de mejor manera el funcionamiento del mismo; y de esta manera al momento de lidiar con el cliente, tuve las herramientas necesarias para aclarar y explicar a los usuarios si los escenarios que ellos probaban eran los correctos o no.

También de manera proactiva encontré errores en el sistema, antes de que el cliente se percatara de los mismos. Informé a la gente de desarrollo que arreglaran el problema con el fin de que cuando los clientes probaran dicho escenario, la funcionalidad fuera la correcta. Colaboré con los expertos en soluciones para verificar la viabilidad de que los nuevos requerimientos que el cliente pedía pudieran ser implementados basados en el desarrollo realizado anteriormente y brindé apoyo en la creación de los casos de prueba a ser ejecutados para verificar que los nuevos requerimientos a ser implementados en el sistema cumplan con lo especificado.

Una de mis tareas principales fue redactar el procedimiento de las instalaciones de los diversos componentes del sistema, explicando qué versión se iba a instalar, el impacto de los cambios a realizarse, los pasos a seguir para la instalación y el tiempo estimado que tomó todo el proceso; con el fin de que el equipo de operación y mantenimiento lo tuviera como referencia para sus instalaciones en el ambiente de producción.

Una vez que se incorporaron nuevos integrantes al equipo, les ayudé a entender el funcionamiento general del sistema y de los módulos customizables correspondientes. Así como también les mostré los componentes y documentos a revisar en caso de que el cliente reporte una incidencia (checar documentación, reportes, conexiones de puertos y logs de información).

Incluso se presentó la oportunidad de realizar una presentación comercial acerca de los diferentes componentes del sistema de pagos móviles a clientes potenciales interesados en adquirir y adaptar el producto.

La buena respuesta que ha tenido el sistema de pagos móviles en México ha propiciado que los clientes vean una gran oportunidad de negocio implementar dicha solución en otros países de América Latina. Usando como base la arquitectura y diseño utilizado en México, se planea implementar a corto y largo plazo la solución de pagos móviles con la confianza de que el producto se acoplará de una manera sencilla a las necesidades particulares de las nuevas entidades a ser integradas, sin perder la calidad en los servicios ofrecidos.

Conclusiones

La participación durante todo el ciclo de vida del proyecto me ha ayudado enormemente a mi crecimiento profesional, dado que nunca había colaborado en un proyecto de esta magnitud e importancia. Ésta ha sido mi primera experiencia profesional y el aprendizaje ha sido basto en muchos aspectos técnicos; además de que se ha desarrollado considerablemente mi capacidad de análisis y resolución de problemas.

No ha sido un proyecto fácil por diversos factores: los clientes, que en este caso son los bancos, son bastante duros, demandantes y siempre exigen mejoras en los tiempos de resolución de problemas; trabajar con desarrolladores y analistas de otros países (en este caso de Israel, India y Francia) fue complicado por obvias barreras culturales y de lenguaje que con el tiempo se fueron reduciendo poco a poco; además que desde el primer día en la empresa fue necesario que trabajara en las oficinas del cliente sin tener un esquema definido de trabajo, sin la documentación necesaria, sin el equipo necesario y sin una capacitación previa.

Conforme ha pasado el tiempo dentro de la empresa, y en particular dentro del proyecto de pagos móviles, las labores y actividades que me fueron asignadas se volvieron más extensas y complejas, pero en general la calidad de trabajo se ha hecho notar y de cierta manera me he ganado el respeto tanto de mis compañeros de trabajo como de los clientes; aunque esta relación, por motivos obvios ha sido complicada, al final del día la retroalimentación ha sido positiva y se tiene un buen nivel de confianza en la eficiencia de solución de problemas.

Pero con el transcurso del tiempo, el equipo de trabajo fue mejorando y la relación con los clientes y con el equipo de desarrollo fue prosperando a tal punto que se logró el objetivo de obtener la autorización de la comisión reguladora para que el producto saliera a producción. Como consecuencia, esta tecnología innovadora ya está disponible para el público en general y se tiene contemplado añadir mejores para atraer a una nueva gama de usuarios.

Glosario

- API - Interfaz de programación de aplicaciones
- ASCII - Código Estándar Estadounidense para el Intercambio de Información
- ATM – Cajero automático
- EAR – Archivo Enterprise
- EJB - Enterprise JavaBeans
- GNU - GNU no es Unix
- GPS - Sistema de posicionamiento global
- GSM - Sistema global para las comunicaciones móviles
- HTTP - Protocolo de transferencia de hipertexto
- HTTPS – Protocolo seguro de transferencia de hipertexto
- HA – Alta disponibilidad
- IP – Protocolo de Internet
- ISO - Organización Internacional de Normalización
- IVR - Respuesta de voz interactiva
- javac – Compilador Java
- javaws – Java Web Start
- JMS – Servicio de mensajes Java
- LTE - Long Term Evolution
- LDAP - Protocolo Ligero de Acceso a Directorios
- NAT – Conversión de direcciones de red
- POS – Punto de venta
- RPM - Red Hat Package Manager

- SID – Oracle System ID
- SQL - Lenguaje de consulta estructurado
- SSL - Capa de conexión segura
- TCP - Protocolo de control de transmisión
- TPS – Transacciones por segundo
- UDDI – Descripción, descubrimiento e integración universal
- USSD - Servicio Suplementario de Datos no Estructurados
- VPN - Red privada virtual
- WAR – Archivo de aplicación web
- WSDL - Lenguaje de Descripción de Servicios Web
- XML - Lenguaje de marcas extensible
- XSD – Definición de esquema XML

Referencias

- Oracle WebLogic Server 11g: Administration Essentials
- http://juliorestrepo.files.wordpress.com/2010/08/redes-de-computadoras-tanenbaum-4ta-edicion-espanol.pdf
- http://www.w3schools.com/soap/soap_intro.asp
- http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=31628
- http://msaffirio.wordpress.com/2006/02/05/%C2%BFque-son-los-web-services/
- http://www.iec.csic.es/criptonomicon/java/quesjava.html
- http://personales.upv.es/rmartin/TcpIp/cap02s10.html
- http://ezetina.wordpress.com/2010/01/12/caracteristicasunix/

- http://www.belgrano.esc.edu.ar/matestudio/carpeta_de_access_introduccion.pdf
- http://www.maestrosdelweb.com/editorial/tutsql1/
- http://blog.jotadeveloper.com/2009/01/22/que-es-jpos/
- http://neo.lcc.uma.es/evirtual/cdd/tutorial/aplicacion/http.html
- http://www.iec.csic.es/criptonomicon/ssl.html
- http://img.redusers.com/imagenes/libros/lpcu097/capitulogratis.pdf
- http://www.financialtech-mag.com/000_estructura/index.php?id=24&idb=183&ntt=11296&sec=10&vn=1
- http://www.programacion.com/articulo/bea_weblogic:_introduccion_129
- http://www.eveliux.com/mx/la-evolucion-de-la-telefonia-movil.php
- http://www.eluniversal.com.mx/finanzas/91029.html
- http://eleconomista.com.mx/economia-global/2012/04/19/sin-acceso-bancos-mas-75-las-personas-bm
- http://fedora2014.blogspot.mx/2012/05/servicio-transfer-de-telcel-disponible.html
- www.cincodias.com/articulo/opinion/sistemas-core-bancarios/20090311cdscdiopi_5/
- http://soporte.epoint.es/index.php?_m=knowledgebase&_a=viewarticle&kbarticleid=134
- http://www.cicei.com/ocon/gsi/tutorial_perl/cap1.htm
- http://www.wirelessdevnet.com/channels/sms/features/sms.html
- http://tagtag.com/wapmobilesites/enter/information/que_es_umts_
- http://www.slideshare.net/GilbertoIbarra/pruebas-de-estres
- http://es.kioskea.net/contents/271-nat-conversion-de-direcciones-de-red-habilitacion-de-puertos-y

- http://spi1.nisu.org/recop/al01/rmoreno/definicion.html
- http://www.zonacodigos.com/index.php/sql/62-store-procedure
- http://www.tutorialspoint.com/log4j/log4j_quick_guide.htm
- http://histinf.blogs.upv.es/2012/12/03/smartphones/#velnand
- http://historico.elpais.com.uy/suple/economiaymercado/13/03/11/ecoymer_701530.asp
- http://www.4gamericas.org/index.cfm?fuseaction=page&sectionid=249
- http://www.oecd.org/dev/americas/42825480.pdf
- https://iessanvicente.com/colaboraciones/oracle.pdf
- http://www.jorgesanchez.net/bd/arquOracle.pdf
- http://www.techopedia.com/definition/15777/red-hat-enterprise-linux-rhel
- http://www.computerworld.es/tendencias/que-es-el-balanceo-de-carga
- http://www.if-not-true-then-false.com/2010/install-sun-oracle-java-jdk-jre-7-on-fedora-centos-red-hat-rhel/
- http://www.badenas.com/?p=43
- http://jmeter.apache.org/
- http://es.kioskea.net/contents/708-uso-de-telnet
- http://www.nicolasventre.com/?p=216
- http://www.web-manual.net/linux-3/installing-oracle-weblogic-10-3-5-0-on-redhat-linux/attachment/weblogic12/
- http://www.jtech.ua.es/j2ee/2003-2004/abierto-j2ee-2003-2004/ejb/sesion01-apuntes.htm
- http://msaffirio.wordpress.com/2006/02/05/%C2%BFque-son-los-web-services/

- http://es.kioskea.net/contents/300-lenguajes-de-programacion-ndash-api
- http://www.kuriositaet.de/iso8583/introduction.html
- http://www.chileoffshore.com/es/interesting-articles/115-todo-sobre-iso8583
- http://www.oracle.com/webfolder/technetwork/tutorials/obe/fmw/wls/10g/r3/appdeploy/configure/jms/conf_jms_wls.htm
- http://everac99.wordpress.com/2007/11/28/el-oracle-rac-que-es-y-como-funciona/
- http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=introduccion-soapUI
- http://telconomia.com/mercado-movil-en-mexico-al-primer-trimestre-2015
- http://blog.aplicacionesmovil.com/aplicaciones-celular/5-razones-para-el-desarrollo-aplicaciones-moviles/

Anexos

Mensajes ISO 8583

Campos típicos de un mensaje ISO8583

Un mensaje típico podría incluir los siguientes campos:

MTI - Indicador de tipo de mensaje.

Bit 2 - Número de cuenta primaria

Bit 3 - Código de procesamiento

Bit 7 - Fecha y hora de la transmisión

Bit 11 - Código de auditoría

Bit 12 - Hora local de transacción

Bit 13 - Fecha local de transacción

Bit 32 - Código de identificación de la institución de adquisición

Bit 38 - El código de autorización como respuesta

Bit 39 - Código de respuesta

Bit 49 - Código de moneda de la transacción

Diferentes formatos de mensajes ISO8583

Un sistema de tipo switch, que rutea (route) transacciones desde POS/ATM o WEB hacia diferentes bancos y/o instituciones financieras, debe ser capaz de procesar diferentes tipos de ISO8583. Existen varios tipos de formato de intercambio de mensajería ISO8583, pero se pueden clasificar en dos grupos: en ASCII (*American Standard Code for Information Interchange*) o en Binario.

Para un ISO8583 como este:

[000] [0200]

[002] [1234567890123456]

[007] [20100609173030]

[022] [ABC123]

[063] [012345678901234567890123456789012345678901234567890\

12345678901234567890123456789012345678901234567890123456789]

Mensaje ISO8583 en base de ASCII

En formato ASCII, este mensaje ISO8583 se ve así:

00000000H 30 32 30 30 34 32 30 30 30 34 30 30 30 30 30 30	0200420004000000
00000010H 30 30 30 32 31 36 31 32 33 34 35 36 37 38 39 30	0002161234567890
00000020H 31 32 33 34 35 36 30 36 30 39 31 37 33 30 33 30	1234560609173030
00000030H 41 42 43 31 32 33 20 20 20 20 20 20 31 30 30 30	ABC123……1000
00000040H 31 32 33 34 35 36 37 38 39 30 31 32 33 34 35 36	1234567890123456
00000050H 37 38 39 30 31 32 33 34 35 36 37 38 39 30 31 32	7890123456789012
00000060H 33 34 35 36 37 38 39 30 31 32 33 34 35 36 37 38	3456789012345678

00000070H 39 30 31 32 33 34 35 36 37 38 39 30 31 32 33 34	9012345678901234
00000080H 35 36 37 38 39 30 31 32 33 34 35 36 37 38 39 30	5678901234567890
00000090H 31 32 33 34 35 36 37 38 39 30 31 32 33 34 35 36	1234567890123456
000000a0H 37 38 39	789

Este tipo de mensaje ISO8583 es más fácil de transportar desde un ambiente a otro, también se llamas zonas. A diferencia de los ISO8583 de tipo binario, el que muchas veces debe ser codificado para ser transportado a otras zonas. La desventaja de los mensajes en base ASCII es que el largo total del mensaje es más largo que el de los binarios.

Mensajes ISO8583 en base binario

El mismo mensaje ISO8583, en formato binario, se ve así:

00000000H 02 00 42 00 04 00 00 00 00 02 16 12 34 56 78 90	..B.....4Vx.
00000010H 12 34 56 06 09 17 30 30 41 42 43 31 32 33 20 20	.4V...00ABC123..
00000020H 20 20 20 20 01 00 30 31 32 33 34 35 36 37 38 39	0123456789
00000030H 30 31 32 33 34 35 36 37 38 39 30 31 32 33 34 35	0123456789012345
00000040H 36 37 38 39 30 31 32 33 34 35 36 37 38 39 30 31	6789012345678901
00000050H 32 33 34 35 36 37 38 39 30 31 32 33 34 35 36 37	2345678901234567
00000060H 38 39 30 31 32 33 34 35 36 37 38 39 30 31 32 33	8901234567890123
00000070H 34 35 36 37 38 39 30 31 32 33 34 35 36 37 38 39	4567890123456789
00000080H 30 31 32 33 34 35 36 37 38 39	0123456789

Claramente, el largo del mismo mensaje ISO8583 en formato binario es más corto que en formato ASCII. Para utilizar el formato binario, existe una exigencia de "padding". En el ejemplo anterior, el campo 63 es tipo LLLVAR (Variable de prefijo longitud 3), por lo tanto en formato ASCII se comienza por "1000123456...." donde 100 es el largo, y la cadena es "0123456....." corresponde al dato. En formato binario, el tipo LLLVAR se convierte a "01 00" que realiza un padding de un cero a la izquierda del campo largo, así se convierte como LLLLVAR.

Creación de cola JMS en Weblogic (Consola de administración)

1. Acceder a la consola de administración y seleccionar *Servidores JMS* en la sección de *Configuraciones de servicios de dominio*; luego seleccionar *Editar y Editar* en la sección de *Centro de cambios*
2. Dar click en *Nuevo* en la tabla *Servidores JMS*
3. Llenar la información de los siguientes campos y dar click en Siguiente:
   - Nombre: testJMSServer
   - Almacenamiento permanente: Ninguno
4. Seleccionar el servidor objetivo para desplegar el servidor JMS
5. Seleccionar *Activar cambios* para guardar la configuración
6. Navegar en la consola hacia servidor>Servicios>Mensajería>JMS. Seleccionar *Bloquear y editar.*
7. Seleccionar *Nuevo* en la tabla Módulos *JMS* y llenar las propiedades:
   - Nombre: testJMSModule
   - Descriptor del nombre: testJMSModule
8. Seleccionar un servidor manejado como objetivo y seleccionar Siguiente
9. Seleccionar el checkbox que dice: *Te gustaría agregar un recurso al módulo JMS?* y dar *Finalizar*
10. En *Configuraciones* de testJMSModule, seleccionar *Subdespliegues*
11. En *Subdespliegues*, crear uno nuevo con su respectivo nombre (testSubdomule) y dar click en Siguiente
12. Seleccionar el servidor objetivo (en este caso testJMSServer) y finalizar
13. En las configuraciones de testJMSModule, seleccionar Nuevo para crear una nueva cola JMS en el módulo JMS
14. Seleccionar *JMS System Module Resource*, dar click en *Cola* y siguiente
15. Llenar los siguientes parámetros de y dar click en siguiente:
   - Nombre: testJMSQueue
   - Nombre de JNDI: testJMSQueue
   - Plantilla: Ninguna
16. Seleccionar el subdespligue creado anteriormente en el servidor JMS correspondiente y finalizar.
17. Seleccionar *JMS System Module Resource*, dar click en *Tópico* y siguiente
18. Llenar los siguientes parámetros de y dar click en siguiente:
   - Nombre: testJMSTopic
   - Nombre de JNDI: testJMSTopic
   - Plantilla: Ninguna
19. Seleccionar el subdespliegue creado anteriormente en el servidor JMS correspondiente y finalizar.
20. En el *centro de cambios*, seleccionar *Activar Cambios*
21. Verificar que la cola y el tópico creado aparezcan en el árbol JDNI del servidor

Características de JDK 6 y JDK 7

Java 6

En este sentido, la revisión 6 incorpora un numeroso abanico de mejoras que, si bien pueden considerarse, en su mayor parte, como pequeños avances, en conjunto contribuirán a facilitar considerablemente el trabajo del programador. En el AWT (*Abstract Window Toolkit*) existen dos nuevas clases, llamadas SystemTray y TrayIcon, que hacen posible colocar en el área de notificaciones de la barra de tareas iconos con menús asociados, algo muy habitual en Windows y que también resulta posible en escritorios como GNOME (*GNU Network Object Model Environment*).

Permite crear tanto servicios web como consumidores de dichos servicios, para lo cual incorpora JAX-WS 2.0 (Java API para servicios web XML). La designación de una clase como servicio web se efectúa mediante anotaciones Java, una técnica similar a los atributos que se utilizan en la plataforma Microsoft .NET, tras lo cual es preciso utilizar una herramienta que se encarga de generar, a partir de las anotaciones y el código de la clase, todos los elementos precisos para publicar un servicio web, incluyendo la descripción WSDL, que es el estándar a la hora de informar a los potenciales consumidores sobre las características del servicio que se ofrece.

Se ha implementado la posibilidad de generar contenido web en un servidor Java directamente desde un guion, siendo preciso utilizar código en este lenguaje previamente compilado, gracias a una API abierta basada en la especificación JSR 223, que también hace posible embeber motores de scripting dentro de aplicaciones Java.

Otras características notables son: Apache Derby, una base relacional de datos implementada enteramente en Java; soporte para Firefox de Java Plug-in y Java Web Start; nueva API que soporta la normalización de texto internacional Unicode; mejoras en las limitaciones de la implementación de los ficheros jar y zip.

Java 7

Se trata de la primera versión de la plataforma Java bajo la dirección de Oracle. Esta versión Java SE 7, es el resultado del desarrollo para el conjunto de la industria que implica revisiones abiertas, versiones provisorias y una intensa colaboración entre los ingenieros de Oracle y los miembros del ecosistema Java en todo el mundo, a través de la comunidad OpenJDK

- Cambios en el lenguaje, para incrementar la productividad del desarrollador y simplificar las tareas comunes de programación disminuyendo la cantidad de

código necesario, aclarando la sintaxis y haciendo que el código pueda leerse más fácilmente. (JSR 334: Project Coin)
- Soporte mejorado, para lenguajes dinámicos (entre ellos: Ruby, Python y JavaScript), lo que da como resultado un aumento considerable del desempeño en la máquina virtual. (JSR 292: InvokeDynamic)
- Una nueva API preparada para múltiples núcleos, que permite a los desarrolladores convertir fácilmente los problemas en tareas que pueden ejecutarse en paralelo en un número arbitrario de núcleos de procesador. (JSR 166: Fork/Join Framework)
- Un completa interfaz de I/O, para trabajar con sistemas de archivo que pueden acceder a una variedad más amplia de atributos de archivo y ofrecer más información cuando ocurren errores. (JSR 203: NIO.2)
- Nuevas características de redes y seguridad.
- Mayor soporte de la internacionalización, incluido soporte para Unicode 6.0.
- Versiones actualizadas de numerosas bibliotecas.

También, viene con un nuevo framework denominado Fork/Join que permitirá a los desarrolladores descomponer problemas más fácilmente y ejecutar tareas en paralelo a través de un número arbitrario de núcleos de procesador; y una completa API de Entrada y Salida NIO.2 para trabajar con sistemas de archivos, acceder a una amplia gama de atributos y ofrecer más información cuando se producen errores.

Oracle RAC

Una de las nuevas características incluidas en el software de Oracle es la creación de clusters de base de datos (llamado cluster de aplicación real – RAC). De acuerdo a Oracle, esta funcionalidad permite disponibilidad 24/7, desempeño y escalabilidad. Adicionalmente a esto se incluye la definición de failover transparente (failover de aplicación transparente – TAF) que utilizan las aplicaciones para sincronizar sus peticiones con el clúster de Oracle sin que éstas se enteren que alguno de los nodos del clúster se ha desconectado (Figura 29).

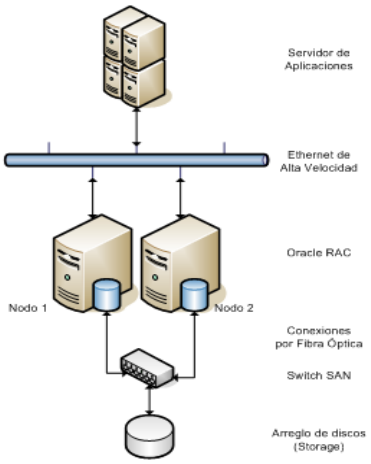

Figura 29. Estructura básica Oracle RAC

Las peticiones a la base de datos son generadas por la aplicación (por ejemplo, desde un pool de conexiones configurado en un servidor de aplicación), y el Oracle RAC en su conjunto es el encargado de direccionar las peticiones al servidor que esté en funcionamiento. Nótese que en esta configuración no existe un balanceo de cargas, por lo que la configuración mostrada es exclusivamente en failover (es decir, todas las peticiones llegarán al Nodo 1, y sólo en caso de que éste deje de funcionar, las peticiones se redireccionarán al Nodo 2).

Adicionalmente, es necesario mencionar que lo que está corriendo en los Nodos 1 y 2 es el listener del motor de base de datos, no la base en sí: la información de la base (los archivos que componen la DB) se encuentra en un arreglo de discos con configuración en espejo para proveer redundancia y por lo tanto, alta disponibilidad.

La principal limitación de Oracle RAC es que no existe un balanceo de carga entre los nodos de base de datos que componen el Oracle RAC: esta configuración por default sólo permite el failover de las peticiones realizadas sobre la base de datos y si el diseño de la arquitectura requiere la optimización de recursos, prácticamente se está desperdiciando la mitad del poder de procesamiento de la capa del back-end.

SOAP UI

soapUI es una aplicación muy versátil que permite probar, simular y generar código de servicios web de forma ágil, partiendo del contrato de los mismos en formato WSDL y con vínculo SOAP sobre HTTP. Esta aplicación tiene dos distribuciones: soapUI freeware (GNU LGPL y opensource java) y soapUIPro (comercial), en versión de escritorio, online y plugin para varios IDE (Integrated Development Environment).

Probar un servicio Web con SoapUI

- Crear un nuevo proyecto soapUI desde el menú File | New soapUI Project (Figura 30)

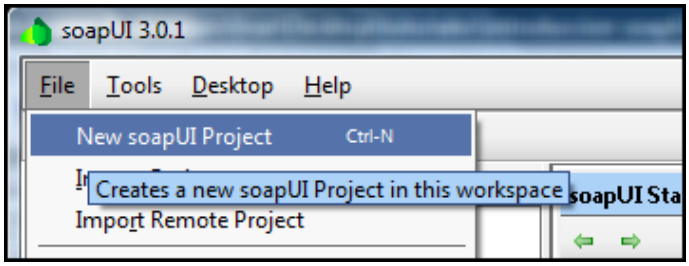

Figura 30. Menú de creación de proyecto en soapUI

- Completar la siguiente información del proyecto:

  Nombre del proyecto

  WSDL/WADL inicial, dirección web o ruta de fichero donde se encuentra el descriptor del servicio web de trabajo.

Crear petición: activar el checkbox de '¿Crear peticiones de prueba para todos los servicios?'. soapUI creará un esqueleto de mensaje SOAP para invocar a los métodos del servicio web

- Si la dirección del descriptor es correcta, soapUI lo recuperará (Figura 31)

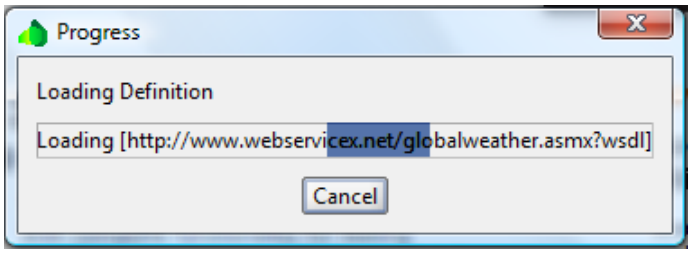

Figura 31. Carga de WSDL en soapUI

- Si no hay errores sintácticos ni ciertas incoherencias semánticas, creará el proyecto con la siguiente estructura (Figura 32):
  Nombre del proyecto
  Interfaces (una o más)
  Dentro de cada interfaz, los métodos del servicio
  Dentro de cada método, un esqueleto de mensaje SOAP, con el nombre genérico Petición 1

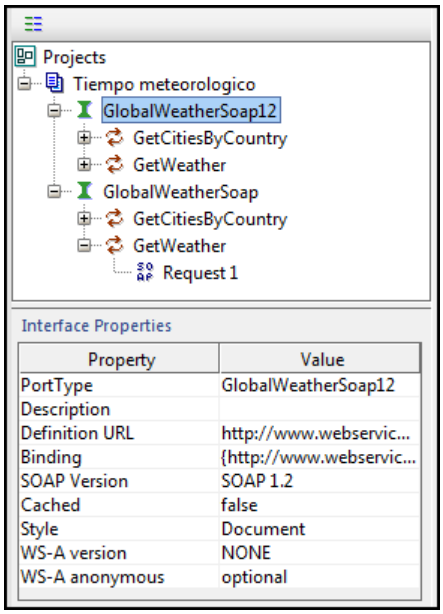

Figura 32. Estructura de proyecto cargado en soapUI

- Haciendo doble click sobre la petición 1, se accede al mensaje de petición SOAP, el que enviaremos al servicio web (Figura 33).

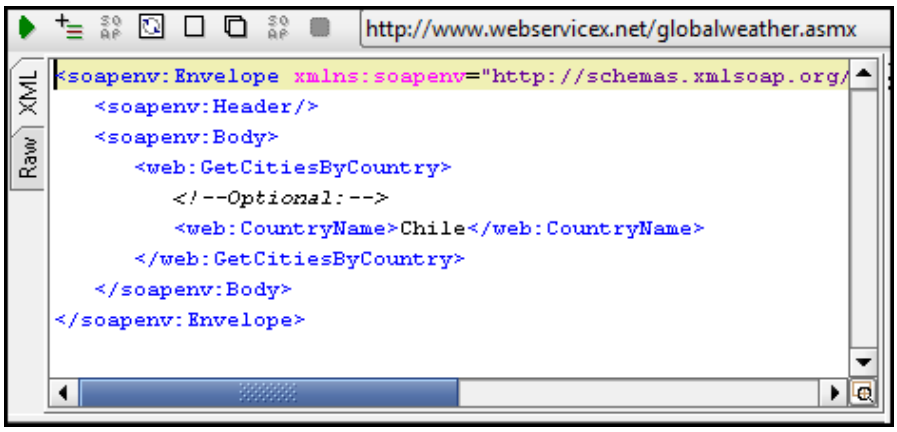

Figura 33. Ejemplo de request SOAP en soapUI

- Enviar el mensaje al servicio pulsando en la fecha verde de la esquina izquierda. Tras unos instantes, se recibe el mensaje de respuesta SOAP que se mostrará a la derecha de la ventana anterior (Figura 34).

Figura 34. Ejemplo de response SOAP en soapUI

- Si se pulsa en la pestaña lateral Raw, se visualiza el mensaje del protocolo HTTP sobre el que viaja SOAP, con el payload y el mensaje en su cuerpo (Figura 35).

Figura 35. Response a nivel HTTP en pestaña Raw