Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Análisis y Diseño de Sistemas 2
Ing. Ricardo Morales
Aux. Kenny Eguizabal

Tarea # 5

GRUPO # 3

Christopher O'Brian Hernández Curruchich 2008-18851
Andrea Virginia Chavarría Guzmán 2009-20081
Ederson Ramírez Hernández 2009-15050
Luis Antonio Mejía Villalta 2009-15383
Luis Fernando Yoc Ávila 2011-23020

Índice

Introducción

SmartNimbus es una empresa que desea ofrecer a sus clientes un servicio para Smartphones, por medio de una aplicación en la cual puedan consultar el saldo del teléfono, duración del contrato que estos poseen, comprar navegación adicional al plan que ya estos poseen, renovación de contrato, promociones que SmartNimbus ofrece y también consultar facturas antiguas, por lo tanto a continuación se presenta una propuesta de modelo para la arquitectura que SmartNimbus podría utilizar para la implementación de las funcionalidades mencionadas anteriormente.

Servicio al Cliente

Requerimientos Funcionales

- Consulta de saldo: El usuario debería ser capaz de consultar el saldo de su teléfono móvil a través de la aplicación.
- Consulta de duración del contrato: El usuario debería ser capaz de consultar el tiempo que hace falta para la terminación de su contrato.
- Compra de navegación adicional a la de su plan: El usuario podrá acceder al plan de pago para comprar más navegación.
- Renovación de contrato: Al terminar el contrato el usuario debería poder acceder a un apartado que le permita leer y aceptar los términos de un nuevo contrato para renovar el que ha vencido.
- Promociones recientes: El usuario tendrá acceso a un apartado que le indique cuales son las promociones que la empresa de telecomunicaciones pública.
- Consulta de facturas: Este apartado permitirá al usuario estar al tanto de cuánto ha pagado por los servicios en el transcurso de su contrato o bien mientras tenga el mismo número telefónico.

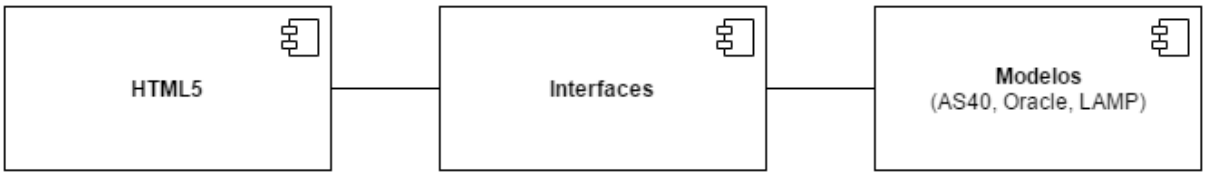
Requerimientos no Funcionales

- Rapidez: Los servicios deben ser accedidos por los usuarios con un 20% del tiempo de respuesta de los servidores básicos (AS400, Oracle y MySQL).
- Seguridad: Los datos deben ser enviados a los servidores básicos encriptados.
- Estabilidad: Se espera que la nueva aplicación esté disponible el 99% del tiempo que los servidores básicos estén funcionando.
- Vinculación: Mantener iniciada la sesión del usuario hasta que él mismo decida salir de ella.

Arquitectura

Patrón de diseño

Debido a las interacciones que los sistemas deben tener con la aplicación de atención al cliente se ha pensado en utilizar el patrón de diseño MVC, siendo así que las interfaces diseñadas para comunicación entre los sistemas serán los controladores, tomando los existentes como el modelo y las vistas serían la aplicación en HTML5.



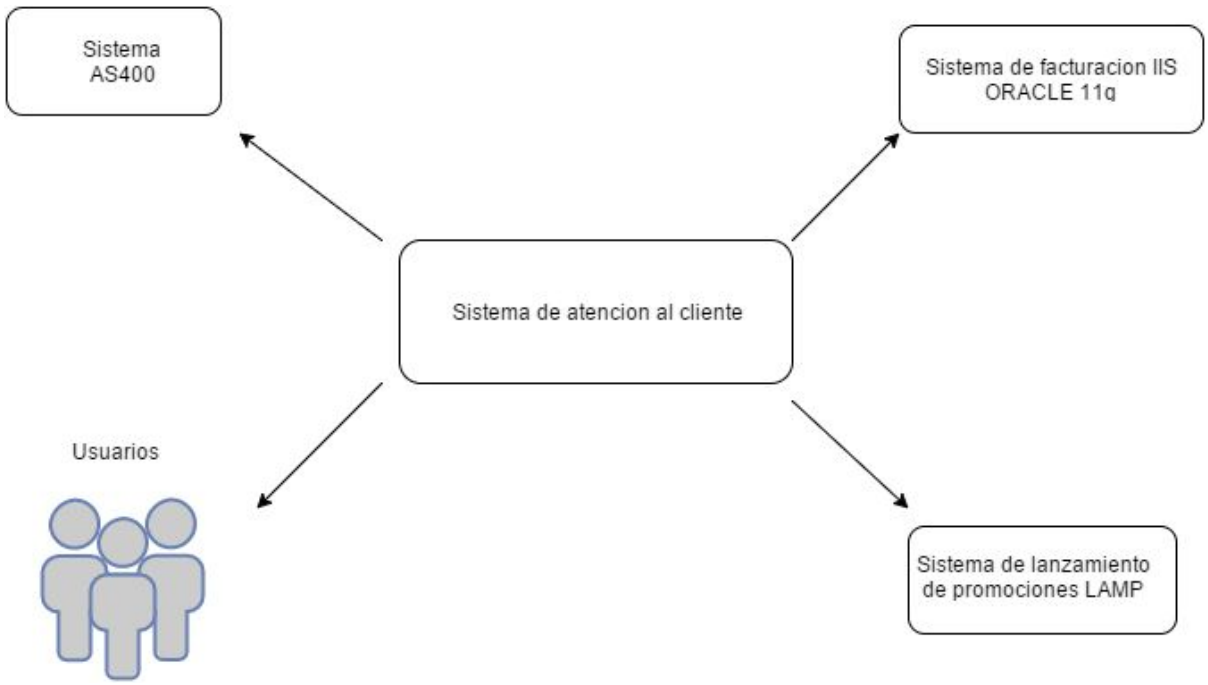
Lenguaje de programación

Ya que se necesita la interacción de varios sistemas y posiblemente varias plataformas se necesita un lenguaje que lo soporte, por lo tanto el equipo de desarrollo ha decidido utilizar Java como el indicado para realizar esta tarea.

Punto de vista de Contexto

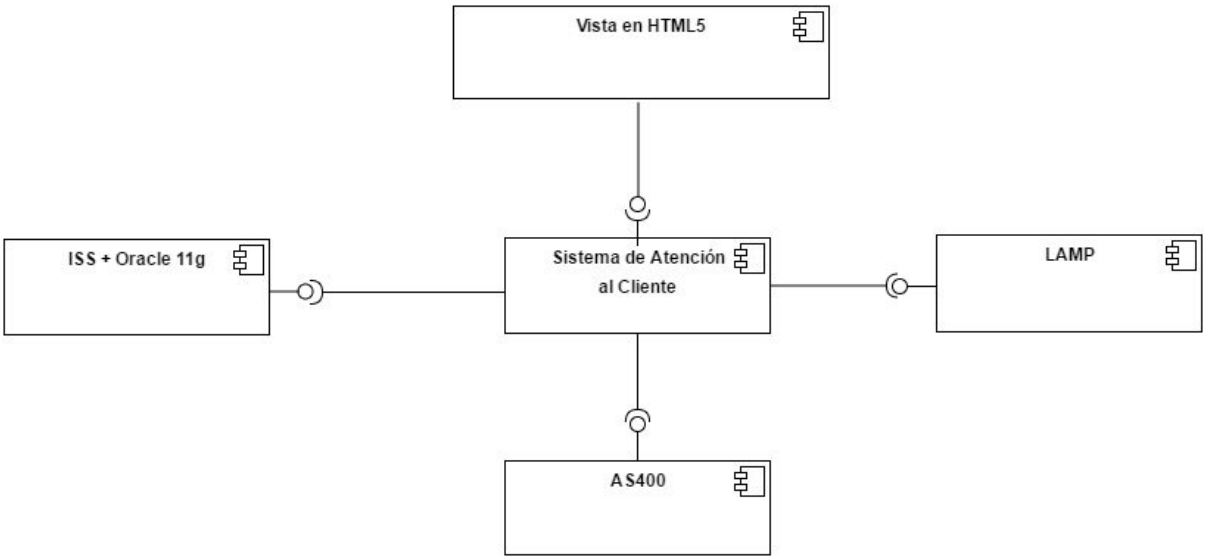
Las unidades externas que interactúan con el sistema de atención al cliente son las siguientes:

- Sistema AS400
- Sistema de facturación IIS y Oracle 11g
- Sistema de lanzamiento de promociones LAMP
- Usuarios



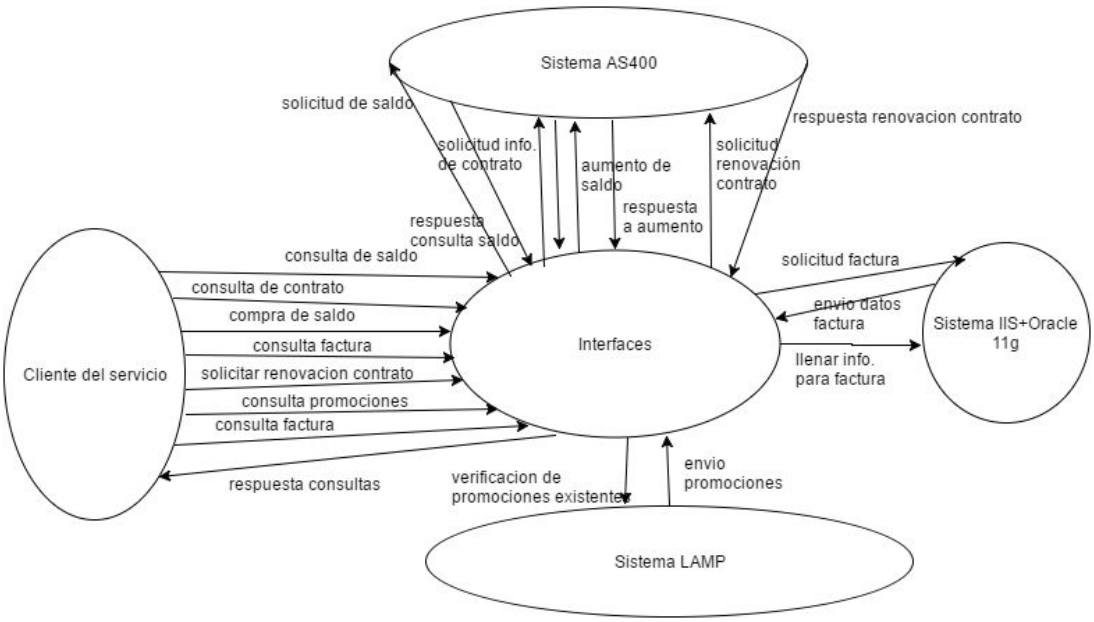
Punto de vista funcional

Se necesitan realizar cuatro interfaces, una para cada parte de los sistemas externos incluyendo la vista en HTML5.

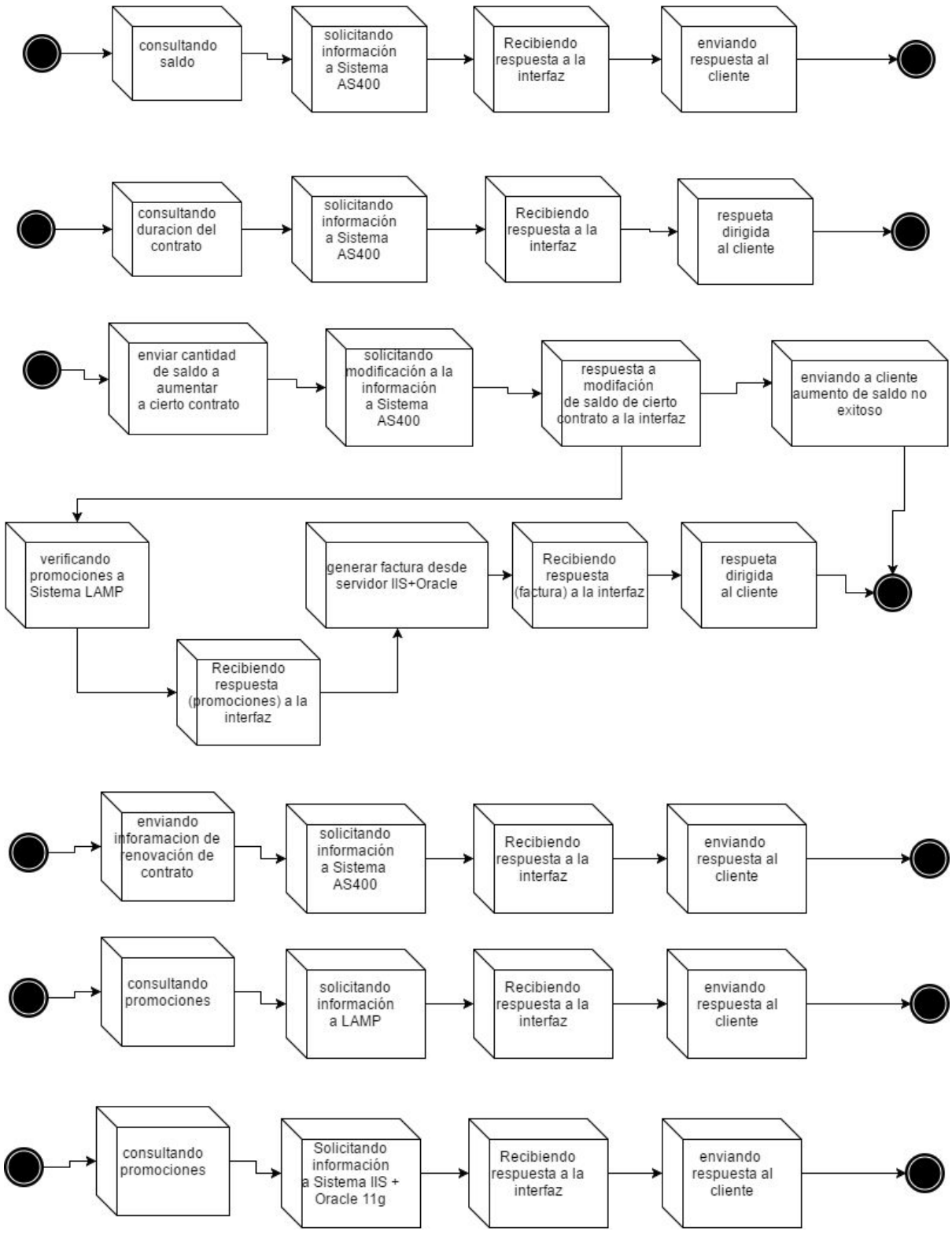


Punto de vista de información

Para entender el flujo de la información que la entidad cliente solicita y la interacción con la interfaz que comunica con los los tres servidores de datos, es necesario usar una Diagrama de Flujo de información, como se puede observar la información viaja en su mayoría como consulta de datos por parte del cliente y la interfaz debe de tratar con el tráfico de dicha información, solicitud y respuesta al cliente. Para simplificar el diagrama, se decidió el uso de una sola respuesta que representa el flujo de todas las respuestas que el cliente recibe a todas sus solicitudes. Aunque con un diagrama de flujo de información no es posible ver el comportamiento de los datos a través del tiempo, por eso mismo se aplica igualmente una serie de diagramas de ciclo de vida de la información.



Para mejor compresión se uso un diagrama para cada tipo de solicitud posible que el usuario realiza, la mayoría con la misma naturaleza en cómo viaja la información, solicitud o modificación de cierto dato, la interfaz mediará con determinado servidor, según sea la naturaleza de la solicitud y así mismo controlará la respuesta de los tres diferentes servidores; en cambio, la complejidad del ciclo de vida de la información requerida para el aumento de saldo de alguno de los contratos de un cliente le será necesario los tres servidores de datos.



Conclusión

Para la propuesta de esta arquitectura se decidió utilizar el patrón de arquitectura MVC (modelo-vista-controlador) el cual separa los datos de la lógica de negocio es decir poder separar las responsabilidades(funcionalidades) dentro de la aplicación y poder tener o aplicar una reutilización de código de ser necesario. En la parte del manejo de información se definió una manera de uso de respuesta-consultas para que así dependiendo de la solicitud del cliente la aplicación se conecte al servidor correspondiente esto para la mayoría de las funcionalidades que se precisan en el aplicativo exceptuando la funcionalidad de saldo en la cual se deberá comunicar a los 3 servidores que se tienen (AS/400, LAMP, IIS+Oracle11g) para poder obtener una respuesta satisfactoria.