# Desarrollo para Sistemas Distribuidos

Equipo: Yii

Repositorio:

<https://github.com/atiliotapia/UPC-SD>

<https://github.com/atiliotapia/coactivosMVCyii>

<https://github.com/jluismoran/proyecto-sd>

Integrantes:

Atilio Tapia - u821143 - github: atiliotapia

Eduardo Escobar - u201121591 - github: u201121591

Jhonathan Moran - u201100743 - github: jluismoran

Daniel Sanchez - u201100470 - github: vidasaor

Horario: C61A

Profesor: Jose Diaz

Ciclo: 2012-02 m1

nota preliminar: 4 puntos

## Índice

[Desarrollo para Sistemas Distribuidos](https://docs.google.com/document/d/1cr99yopC6LZFlOhEu6KYiLc6b6ssN7zQ26ZlruKqJSQ/edit?pli=1#heading=h.t24wdjehsrnc)

[Índice](https://docs.google.com/document/d/1cr99yopC6LZFlOhEu6KYiLc6b6ssN7zQ26ZlruKqJSQ/edit?pli=1#heading=h.db108ghthpbf)

[Introducción](https://docs.google.com/document/d/1cr99yopC6LZFlOhEu6KYiLc6b6ssN7zQ26ZlruKqJSQ/edit?pli=1#heading=h.ast5fg7qqypl)

[Antecedentes](https://docs.google.com/document/d/1cr99yopC6LZFlOhEu6KYiLc6b6ssN7zQ26ZlruKqJSQ/edit?pli=1#heading=h.vdveba4rixnk)

[Objetivo del proyecto](https://docs.google.com/document/d/1cr99yopC6LZFlOhEu6KYiLc6b6ssN7zQ26ZlruKqJSQ/edit?pli=1#heading=h.e7ptcljh7ir1)

[Beneficios del proyecto](https://docs.google.com/document/d/1cr99yopC6LZFlOhEu6KYiLc6b6ssN7zQ26ZlruKqJSQ/edit?pli=1#heading=h.r0nsfnrl0c1a)

[Descripción del proceso](https://docs.google.com/document/d/1cr99yopC6LZFlOhEu6KYiLc6b6ssN7zQ26ZlruKqJSQ/edit?pli=1#heading=h.c5txe4bt48dr)

[Arquitectura del sistema](https://docs.google.com/document/d/1cr99yopC6LZFlOhEu6KYiLc6b6ssN7zQ26ZlruKqJSQ/edit?pli=1#heading=h.11jef75svxhw)

[Desarrollo del sistema](https://docs.google.com/document/d/1cr99yopC6LZFlOhEu6KYiLc6b6ssN7zQ26ZlruKqJSQ/edit?pli=1#heading=h.wwqgbfjqym8e)

[Conclusiones](https://docs.google.com/document/d/1cr99yopC6LZFlOhEu6KYiLc6b6ssN7zQ26ZlruKqJSQ/edit?pli=1#heading=h.3424w6c8xq6)

### Introducción

El proyecto debe solucionar los problemas de de cobranzas en la municipalidad de San Juan de Miraflores mediante soluciones que apliquen sistemas distribuidos, debido a que el Sistema de Cobros del municipio y el Sistema de Cobranza de la empresa contratada para realizar los cobros están en diferentes lenguajes (PHP y Java).

### Antecedentes

La municipalidad de San de Juan de Miraflores cuenta con sistemas actualmente que no esta en la nube y el departamento de Coactivos quienes realizan las cobranzas de los clientes morosos necesita llevar un control del todo los documentos entregados saldos de cobranzas. Para este fin se conectará el sistema de cobros del municipio en lenguaje PHP con un servicio web en Java de la empresa encargada de ejecutar los cobros.

### Objetivo del proyecto

El objetivo principal es que la Municipalidad de San Juan de Miraflores lleve un mejor control sobre las cobranzas de sus contribuyentes. Para ello utilizaremos diversas plataformas en el acceso del sistema de cobranzas a través de La Nube.

### Beneficios del proyecto

Los beneficios de implementar una solución con arquitectura orientada a servicios (SOA) es que posee una amplia gama de características y funcionalidades que la plataforma satisface todos los requisitos tecnológicos en una SOA para acceder a los datos, limpiarlos, transformarlos y entregarlos en el momento adecuado, es decir, cuando, donde y como su negocio lo necesite. La plataforma es el complemento perfecto para las tecnologías existentes en una SOA.

**Compatibilidad flexible con la integración de datos**

* Ofrece servicios de datos escalables y flexibles que cubren todo el ciclo de vida de integración de datos:
  + Servicios de infraestructura para la autenticación, el control de acceso y el registro
  + Servicios de acceso para crear, leer y actualizar datos de todas las formas de sistemas back-end, incluido el almacenamiento de datos estructurados, semiestructurados y no estructurados
  + Servicios de integración para ofrecer funciones sofisticadas de integración de datos, como la transformación, la limpieza, el enriquecimiento y la federación
  + Servicios de metadatos para gestionar y utilizar metadatos técnicos y de negocio para la auditoría, el linaje y el análisis de impacto

**Entrega en el momento adecuado de datos integrales y precisos**

* Suministra las herramientas que pueden dirigirse de forma proactiva a los problemas de calidad de datos en todas las fuentes de datos
* Ofrece una amplia gama de conectores para ampliar de forma fluida el acceso a diferentes tipos y formatos de datos
* Ofrece datos integrales y precisos —cuando, donde y como se necesite— mediante la integración proactiva de los procesos y las asignaciones de calidad de datos

**Flexibilidad para crear y cambiar la funcionalidad de forma rápida**

* Ofrece un entorno de desarrollo sin codificación que promueve la creación rápida de servicios de datos sofisticados
* Admite la secuenciación u orquestación de servicios de datos individuales para complementar los servicios y procesos de nivel de negocio o de aplicación, más las transacciones y los flujos de trabajo humanos distribuidos
* Permite diferentes [casos de uso](http://origin-wwwnew.informatica.com/es/solutions/soa/Pages/soa_use_cases.aspx) al admitir mecanismos flexibles de entrega de datos en una plataforma que ofrece datos a una arquitectura orientada a servicios (SOA) en el momento adecuado: modos batch, de captura continua de cambio de datos y en tiempo real
* Ofrece información crítica para el negocio a una variedad de usuarios de datos en el formato deseado
* Admite una gestión y un análisis sofisticados de metadatos para el linaje de datos, la reconciliación de los metadatos empresariales y el entendimiento de las interdependencias entre los datos y los procesos
* Suministra una tecnología complementaria para la integración de infraestructuras de IT empresariales y [centros de competencia de integración](http://origin-wwwnew.informatica.com/es/solutions/integration_competency_centers_icc/Pages/index.aspx)

**Escalabilidad de la plataforma**

* Ofrece el rendimiento y la escalabilidad necesarios para gestionar grandes volúmenes de tareas de integración de datos complejos compatibles con operaciones de misión crítica las 24 horas del día, los siete días de la semana
  + División de los datos integrados para el procesamiento paralelo de varias CPU
  + Compatibilidad con sistemas grid distribuidos y de varios nodos
  + Alta disponibilidad, tolerancia de fallos y failover
* Se amplía de forma inteligente y automática para admitir más solicitudes de servicios de datos al mismo tiempo

### Descripción del proceso

Modele el proceso que desea automatizar. Identifique actividades, reglas de negocio y flujo de secuencia.

Elabore un diagrama del proceso utilizando UML o BPMN así como la lista de historias de usuario.

Proceso de Cobranza

Se inicia cuando el cliente mantiene una deuda pendiente con la Municipalidad de San Juan de Miraflores. En primer lugar, se ingresa el cliente que tiene un pago por cancelar, a este ingreso se le adjunta el documento coactivo

### Arquitectura del sistema

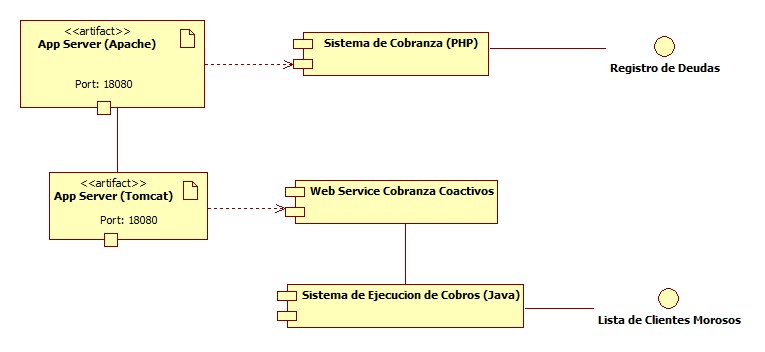
La arquitectura del sistema de cobranza distribuido se compondrá de los siguientes artefactos y nodos.

***Artefactos***

* + Servidor de Aplicaciones Apache .- Aquí se despliega la aplicación de cobranzas, en PHP, del municipio.
  + Servidor de Aplicaciones Tomcat .- Aquí se despliega el sistema de la empresa encargada de ejecutar los cobros.

***Nodos***

* + Sistema de Cobranza del Municipio .- Se registra las personas naturales o jurídicas y sus deudas con la municipalidad.
  + Sistema de Ejecución de Cobros .- Donde la empresa encargada de realizar los cobros obtendrá la información de la base de datos de la municipalidad.
  + Web Service de Cobranza de Coactivos .- Implementará los métodos y los servicios para obtener la lista de clientes morosos de la base de datos de la municipalidad y esta pueda ser consultada por otros sistemas.



### Desarrollo del sistema

Sprint 1

* Desarrolle la capa de presentación del sistema.

- Se hizo la ventana de validación

- Se desarrolló el sistema de cobros en el framework Yii (php)

- Se subió el proyecto en el repositorio de Github

- Se elabora el documento.

- Se instruyó al grupo sobre la utilización o manuales para hacer una inducción del framework yii.

<http://www.youtube.com/playlist?list=UU03XybngBmefq84tQKoQD5w&feature=playlist-comment>

- Se decidió utilizar el lenguaje java para hacer un programa que liste

la cobranza de morosos.

- Se sube el trabajo al siguiente repositorio.

**Mockups**

<https://github.com/atiliotapia/coactivosMVCyii>

[www.aesoluciones.com/clientes/coactivos](http://www.aesoluciones.com/clientes/coactivos)

Usuario: aldo

Clave: aldo

SCRUM del Proyecto

<http://scrumy.com/seam82boxed>

1.1 Capas y niveles

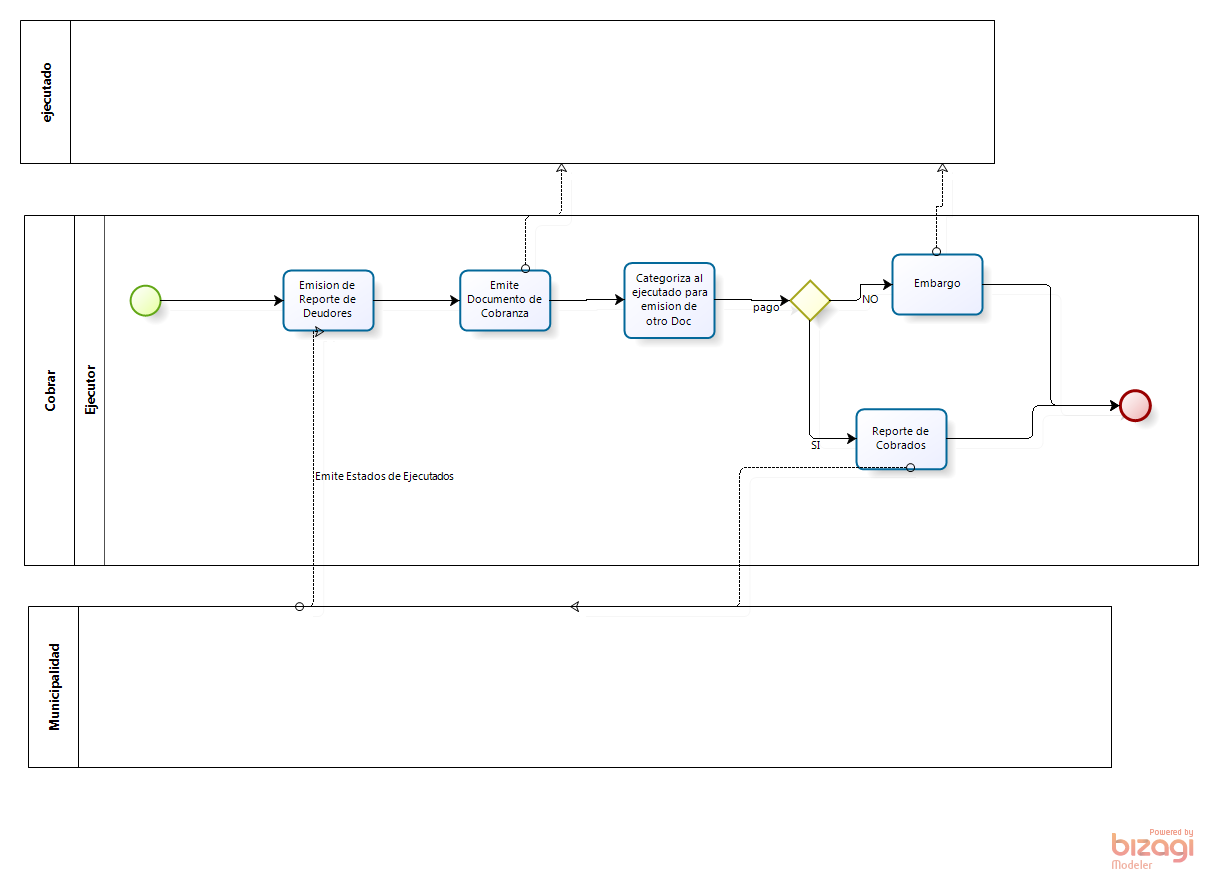
1. Capa de presentación: es la que ve el usuario (también se la denomina "capa de usuario"), presenta el sistema al usuario, le comunica la información y captura la información del usuario en un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). También es conocida como interfaz gráfica y debe tener la característica de ser "amigable" (entendible y fácil de usar) para el usuario. Esta capa se comunica únicamente con la capa de negocio.
2. Capa de negocio: es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él. También se consideran aquí los programas de aplicación.
3. Capa de datos: es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio. Todas estas capas pueden residir en un único ordenador, si bien lo más usual es que haya una multitud de ordenadores en donde reside la capa de presentación (son los clientes de la arquitectura cliente/servidor). Las capas de negocio y de datos pueden residir en el mismo ordenador, y si el crecimiento de las necesidades lo aconseja se pueden separar en dos o más ordenadores. Así, si el tamaño o complejidad de la base de datos aumenta, se puede separar en varios ordenadores los cuales recibirán las peticiones del ordenador en que resida la capa de negocio

* Desarrolle los Web Services SOAP identificados.
* Desarrolle tests de unitarios y de integración automatizados para todos los servicios desarrollados.

Sprint 2

* Desarrolle la capa de presentación e integración del sistema.
* Desarrolle todos los servicios identificados (Web Services SOAP, Web Services REST y mensajería)
* Desarrolle tests de unitarios y de integración automatizados para todos los servicios desarrollados.

HISTORIA DE USUARIO



### Conclusiones

Elabore las conclusiones del proyecto (mínimo 4).