



Universidad Tecnológica Nacional
Facultad Regional Buenos Aires
Ingeniería en Sistemas de Información

AÑO 2017

Materia: Diseño de Sistemas

Código de Materia: 082028

Curso: K-3051

Docentes:

Mur, Pablo
Oliva, Miguel
Procopio, Demian
Rico Mendoza, René
Sosa, Ezequiel
Valido, Leandro

Trabajo Práctico: “¿Donde invierto?”

Tipo: Grupal

GRUPO N° 12	
NOMBRE Y APELLIDO	LEGAJO N°
Lucas, Amoroso	151687-5
Nobile, Nicolas	150301-7
Gabriel Figueroa	131972-3
Daniel Adrian Avila	141979-1

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

--	--

Fecha prevista de entrega: **16/ 05/ 2017**

Fecha real de entrega: **16/ 05/ 2017**

Calificación..... Firma.....

Versión <1.0>

Historia de revisión

Fecha	Descripción	Autor	Versión
16/05/2017			1.0

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Tabla de Contenidos

Entrega 1	4
Enunciado	4
Desarrollo	9
Punto 1	9
Diagrama de Casos de uso	9
Interfaz gráfica	10
Diagrama de clases	11
Punto 3	12
Entrega 2	12
Enunciado	12
Desarrollo	14
Gramática	14
Indicadores	14
Cuentas	14
Diagrama de clases	14
Diagrama de Secuencia	15
Entrega 3	16
Enunciado	16
Desarrollo	18
Cargar Empresas y cuentas	18
Cargar Indicadores	18
Menú Metodologías	18
Diagrama de clases	18
Entrega 4	19
Enunciado	20
Desarrollo	22
DER	22

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Diagrama de Clases	24
Entrega 5	24
Enunciado	26
Consideraciones	27
Arquitectura	27
Diagrama de despliegue	28
Diagrama de componentes	29
Diagrama de persistencia	29
Diagrama de protocolos de Red	31
Diagrama de interfaces con sistemas internos y externos	31
Diagrama de clases	32
Diagrama de entidad de relación	34
Entrega 6	34
Enunciado	35
Punto 1	36
Punto 2	36
Punto 3	37
Punto 4	38

Inicialización


- Hacer “mvn install” del contenido de la carpeta modelo, esto generará un .jar que es lo que utilizara las dependencias pom de los otros modelos
- Generar un usuario “dds” con la contraseña “dds” en mysql
- Dirigirse al modelo desktop, paquete “interfaz gráfica” , en la clase main y ejecutarla. Esto generará el schema “dds” y cargará la base con los datos provenientes de los distintos archivos

Luego de realizados estos pasos, se podrán ejecutar los test (mvn test habiéndose dirigido a la carpeta modelo).

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Dónde invierto?	Grupo: 12 – Versión 1.0

Entrega 1

Enunciado



UTNBA
DPTO. INGENIERÍA EN SISTEMAS DE INFORMACIÓN
CÁTEDRA DISEÑO DE SISTEMAS

¿Dónde invierto?

Primera Parte (Primer Cuatrimestre)

1 Contexto Inicial	1
1.1 Héctor	1
1.2 Sabrina	1
2 Dominio	2
2.1 Cuentas	2
2.2 Indicadores	2
2.3 Metodologías	3
2.3.1 Condiciones y Metodologías definidas por el usuario	3
3 Detalle de la problemática	4
4 Entregas	4
4.1 Primera Entrega: Primera aproximación	4

1 Contexto Inicial

Es Martes, 28 de febrero de 2017, y el calor al mediodía en la casa de Villa Martelli es agobiante. Sólo se escuchan las chicharras del jardín, el cooler de la computadora y a Héctor tipeando frenéticamente desde temprano, describiendo todo lo que sabe sobre el mundo de las acciones en la bolsa de Estados Unidos. ¿Por qué lo hace?

1.1 Héctor

Héctor, de 48 años, empezó a investigar sobre este tema algunos años atrás, por pura curiosidad. Con el tiempo, pasó de ser un mero hobby, a un ingreso extra ocasional: en su tiempo libre, asesora a algunos amigos y conocidos que hacen pequeñas inversiones. Sin embargo, las cosas están cambiando en su vida: la empresa en donde trabaja ha empezado a recortar personal, y ante la amenaza de quedarse sin empleo está evaluando dedicarse tiempo completo a la actividad de asesoramiento.

Pero hay un problema: Héctor recopila y procesa toda la información de forma mayormente manual. Consulta sitios en internet que proveen los datos que necesita, los carga en una planilla de cálculo, y realiza su análisis en parte apoyándose en fórmulas que él mismo cargó, y en parte con papel, lápiz y calculadora. Esto es tedioso y le demanda mucho tiempo, y sabe que le impide escalar su negocio.

Para colmo de males, como siempre está aprendiendo sobre la materia, es frecuente que decida modificar algunas de las fórmulas que usa, agregue nuevas o desestime otras que no le fueron útiles en sus análisis, lo que en más de una ocasión introdujo errores en sus análisis que le significaron pérdidas importantes de tiempo. Afortunadamente siempre se dio cuenta del error antes de dar un mal consejo a sus clientes, pero necesita dar una solución a esta cuestión.

1.2 Sabrina

1 de 5

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0



Charlando el último domingo durante una cena familiar, su sobrina Sabrina, ingeniera recientemente recibida pero ya con algunos años de experiencia en el desarrollo de software, lo convenció de automatizar, en parte, lo que hace: construir una aplicación de escritorio simple que le permita cargar la información actualizada sobre las empresas que analiza, los cálculos que realiza y mostrar sus resultados. En definitiva, construir un (muy simple) Sistema de Apoyo a Decisiones. Ella se ofreció a hacerlo, pero para empezar, le pidió le envíe por correo electrónico el dominio con el mayor nivel de detalle posible.

Y esto nos trae nuevamente al 28 de febrero, día en que Héctor se encuentra escribiendo el documento que está a continuación...

2 Dominio

2.1 Cuentas

Periódicamente (típicamente cada año o cada semestre), las empresas que cotizan en bolsa generan balances públicos, que hablan del estado la empresa, expresados en cuentas.

Ejemplo: estas son algunas cuentas:

- [EBITDA](#)
- [EDS](#)
- [Free Cash Flow](#)
- [Ingreso neto en operaciones discontinuas](#) e [Ingreso neto en operaciones continuas](#)

Existen cientos de cuentas posibles, cada una con su propia fórmula de cálculo. Sin embargo, **para el inversionista poco importa calcularlas**, dado que existen bases de datos públicas que ya nos muestran sus valores en cada período.

Ejemplo: Facebook en el año 2016 la cuenta EBITDA tuvo valor de 14870 millones de dólares, y en año 2015, 8162 millones de dólares. Para el análisis que vamos a hacer el cálculo exacto de esta cuenta no es relevante, sino sólo su valor en cada año.

2.2 Indicadores

Si bien los datos de cuentas son la base de cualquier análisis, vamos a trabajar en su lugar con información más refinada. Para eso utilizaremos **indicadores**, que son cálculos (simples, típicamente sólo algebraicos) sobre las cuentas anteriores, que son útiles para el inversionista pero no siempre son publicados directamente por las compañías.

Ejemplo: un indicador es el ingreso neto, y puede calcularse como la suma de dos cuentas:

$$\text{IngresoNeto} = \text{IngresoNetoEnOperacionesContinuas} + \text{IngresoNetoEnOperacionesDiscontinuas}$$

Los indicadores pueden ser aún más interesantes: pueden ser cálculos que combinen cuentas y otros indicadores.

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0



Ejemplo: el [retorno sobre el capital total](#) (ROE), utiliza a otro indicador: el ingreso neto visto anteriormente.

$$\text{Retorno sobre capital total} = \frac{\text{Ingreso Neto} - \text{Dividendos}}{\text{Capital Total}}$$

Al igual que las cuentas, un indicador puede arrojar diferentes valores en cada período para una cierta empresa, por eso es que nos sirven para evaluarla año a año.

Ejemplo: en 2016, las cuentas de Facebook arrojaron los siguientes resultados:

- Ingreso Neto En Operaciones Continuas: 4.273 millones de dólares
- Ingreso Neto en Operaciones Discontinuas: 0

Por tanto, el indicador Ingreso Neto (la suma de las dos cuentas anteriores) arrojó 4.273 millones de dólares

2.3 Metodologías

Finalmente, los inversionistas, en base a su experiencia y estudio, eligen (o incluso, crean) ciertos indicadores que les resultan significativos, y les imponen ciertas condiciones, que nos dicen si vale la pena invertir en una empresa, o si es mejor invertir en una o en otra.

Al conjunto de estos indicadores y condiciones se los conoce como metodologías (o métodos) de inversión, y son un elemento fundamental y diferencial de este sistema, dado que **nos dan pistas** sobre dónde invertir (¿y dónde no!)¹.

Ejemplo: si bien [la metodología de Warren Buffet](#) no ha sido totalmente sistematizada, una posible interpretación nos dice que utiliza 6 indicadores y les aplica ciertas condiciones.

Resumimos a continuación 4 condiciones que **se pueden evaluar de forma automática**:

- Maximizar ROE: una empresa es mejor que otra si durante los últimos 10 años, su ROE fue consistentemente mejor que el de la otra.
- Minimizar el nivel de deuda: una empresa es mejor que otra si [su proporción de deuda](#) es menor
- Márgenes consistentemente crecientes: vale la pena invertir en una empresa en la que su margen durante los últimos 10 años fue siempre consistente
- Longevidad: sólo vale la pena invertir en empresas con más de 10 años. Además, una empresa es mejor que otra si es más antigua.

2.3.1 Condiciones y Metodologías definidas por el usuario

Al igual que con los indicadores, a Héctor le gustaría tener la posibilidad de crear metodologías al vuelo, sin tener que pedirle a Sabrina que modifique el programa. Las condiciones también deberían poder ser definidas por él, con cierta flexibilidad. Por ejemplo:

- Que un indicador sea mayor o menor a cierto valor, en el último año o durante los últimos N años

¹ Decimos que el sistema nos da pistas, y no que nos da una respuesta taxativa la respuesta a la pregunta de dónde invertir, porque **hay muchos más aspectos que el analista considerará que no son automatizables**, relacionados con su experiencia, el perfil de su cliente, y el contexto.

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Dónde invierto?	Grupo: 12 – Versión 1.0



UTN
DPTO. INGENIERÍA EN SISTEMAS DE INFORMACIÓN
CÁTEDRA DISEÑO DE SISTEMAS

- Que un indicador sea mayor o menor que el de otra empresa
- Que un promedio, mediana o sumatoria de un cierto indicador sea mayor o menor a cierto valor
- Que un indicador sea siempre o casi siempre creciente o decreciente durante un período

3 Detalle de la problemática

Entendido el dominio y el contexto, repasemos y formalicemos a los requerimientos funcionales y no funcionales del sistema:

1. Tendrá un único usuario, Héctor, en su rol de analista de inversiones.
2. Será usada desde una computadora de escritorio
3. Deberá permitir
 - a. la carga de cuentas de empresas, por período
 - b. la carga de indicadores creados por el analista
 - c. para una empresa y un período, la consulta de los valores
 - i. de cuentas
 - ii. e indicadores: se podrán encontrar tanto indicadores creados por el analista, como predefinidos.
 - d. para una empresa y un período, la visualización de gráficos comparativos de indicadores en el tiempo
 - e. la carga de metodologías creadas por el analista
 - f. el análisis de una empresa aplicando metodologías
 - g. el guardado de metodologías e indicadores creados por el analista

4 Entregas

Tras reunirse nuevamente Héctor y Sabrina, decidieron que encararán el proyecto de forma iterativa e incremental. ¿Por qué? Porque por un lado, Héctor necesita algo funcionando lo antes posible, y por otro lado, ambos necesitan validar sus ideas sobre qué debe hacer el sistema.

Por otro lado, además de tener en cada entrega algo funcional, esta metodología de desarrollo le dará a Sabrina la posibilidad de en cada iteración, comprender mejor el dominio a medida que diseña su modelo, y a Héctor, la oportunidad de formalizar y sistematizar lo que sabe su proceso de análisis.

4.1 Primera Entrega: Primera aproximación

Para la primera iteración, es necesario elegir la tecnología para la realización del proyecto, decisión arquitectural de gran impacto en el desarrollo del sistema. Parte fundamental de esta entrega es la elección de una tecnología **y su prueba**.

Alternativas: Existen muchos lenguajes de programación populares, de propósito general y de uso extendido para realizar aplicaciones: Java, C#, Ruby, JavaScript, C, etc.

Sugerencia: a fines didácticos, utilizar un lenguaje de objetos, con una orientación tradicional y tipado estático. Las opciones más comunes con esas características son Java y C#.

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0



Además, es necesario proveer a Héctor una interfaz gráfica que sirva para verificar los principales requerimientos del sistema. El objetivo es hacerlo lo más rápido posible, para encontrar inconsistencias lo antes posible.

Alternativas:

- Desarrollar una interfaz de escritorio
- Desarrollar una interfaz web
- Desarrollar únicamente maquetas HTML o en papel

Sugerencia: normalmente, las interfaces gráficas de escritorio son más simples de construir que sus contrapartidas Web (y Mobile), dado que involucran menos tecnologías y todos los componentes conviven en una misma máquina virtual.

Las maquetas en papel, por otro lado, son excelentes formas de validación de prototipos, por ser muy rápidas de construir², pero tienen la contra de no ser reutilizables y ser mucho menos interactivas.

Para este Trabajo Práctico sugerimos realizar maquetas en papel y al menos construir una pantalla utilizando tecnología de escritorio, a fin de que sirva como prueba tecnológica.

Finalmente, para esta primera iteración se implementarán los siguientes requerimientos:

- Carga de cuentas (3.a)
- Consulta de valores de cuentas (3.b.i.)

Sugerencia: se sabe que Héctor cargará los datos de empresas consultando páginas de Internet. Definitivamente sería una gran mejora si esta carga de datos se realizará de forma automatizada. Sin embargo, para mantener esta iteración simple y dado que Héctor ya estaba haciendo este proceso de carga de forma manual, recomendamos cargar los datos de cuentas de las empresas desde un archivo.

Se pide:

1. Realizar un diagrama de casos de uso que contemple todos los requerimientos descritos [en el punto anterior](#) y los bocetos para las pantallas correspondientes.
 - a. Para pensar: ¿Cómo podría el usuario ingresar y visualizar las fórmulas de los indicadores?
2. Diseñar, implementar y **probar de forma automatizada** el proceso de carga de un archivo con cuentas para varias empresas.

Advertencia: el resultado de este proceso de carga debe dejar a las cuentas de todas las empresas en el ambiente de objetos (en otras palabras: en memoria).

3. Diagramar la arquitectura del sistema

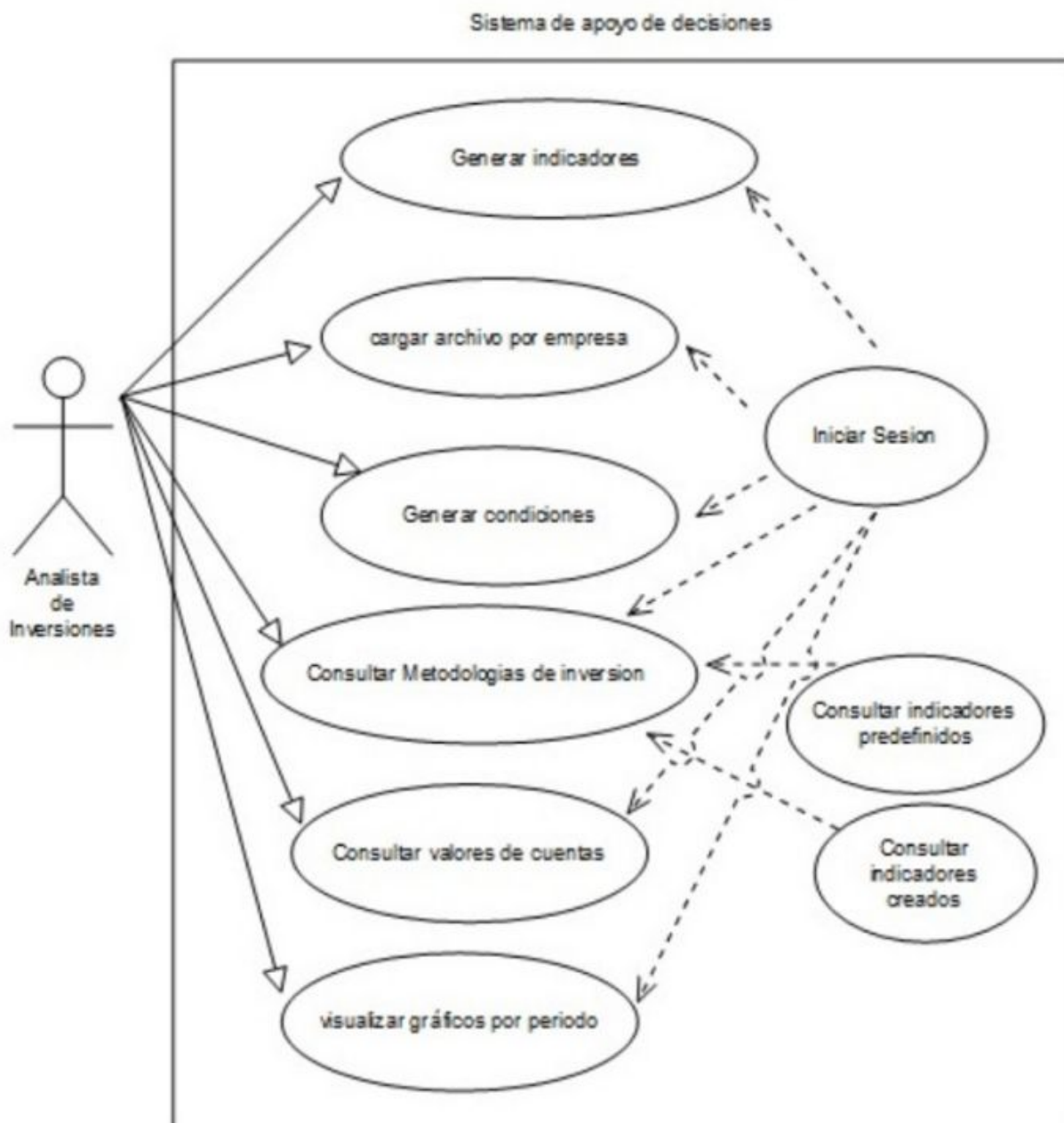
² Un video vale más que mil palabras: <https://youtu.be/GiinoCHKQZY?t=33>

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Desarrollo

Punto 1

Diagrama de Casos de uso



Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Interfaz gráfica

Cargar archivo

Obtener valor de cuenta

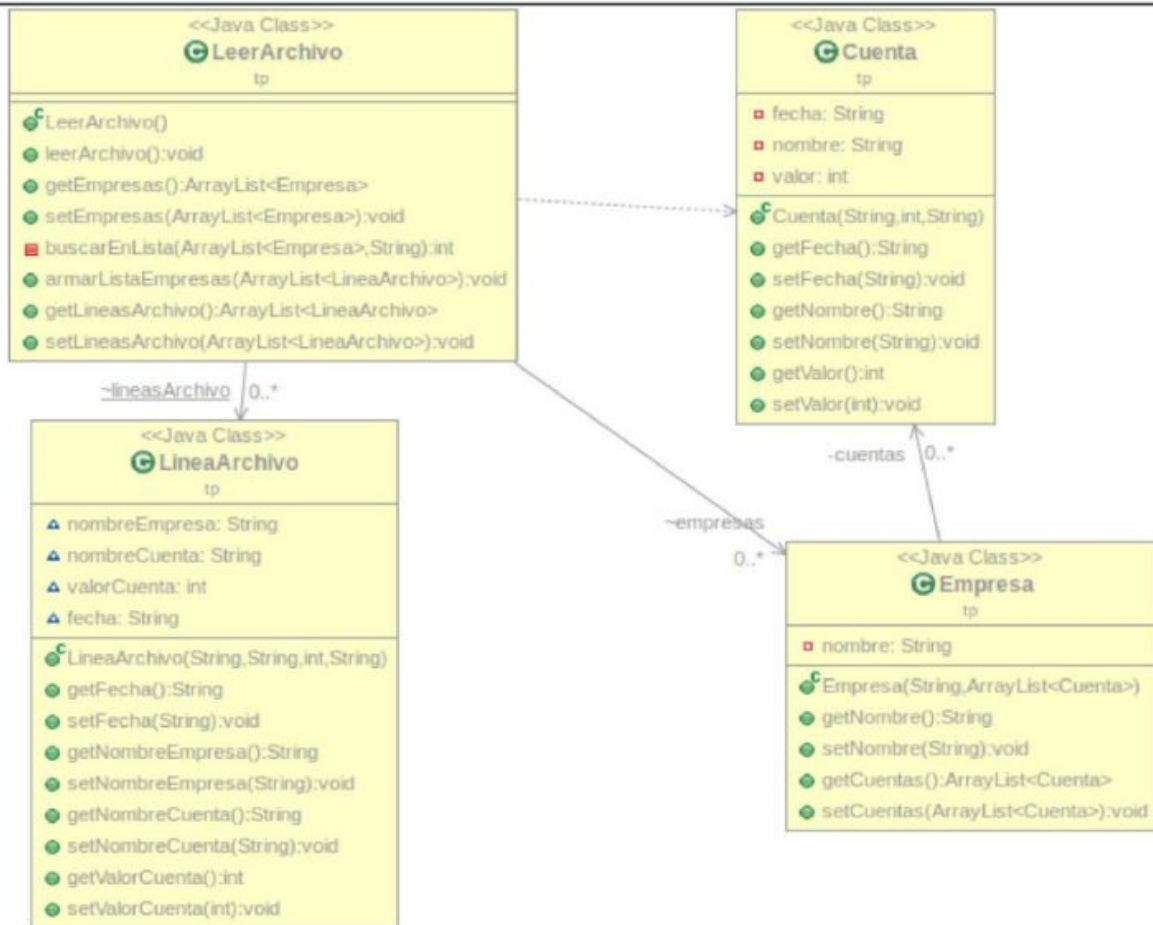
Resultado

Cuenta

Valor

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Diagrama de clases



Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Dónde invierto?	Grupo: 12 – Versión 1.0

Punto 3



Entrega 2

Enunciado

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0



Segunda entrega: Indicadores

En la segunda iteración ampliaremos nuestra aplicación, para dar soporte a los *indicadores*. Recordemos que los indicadores pueden ser tanto:

- **predefinidos**, es decir, estar programados dentro de nuestra aplicación. Esto sirve para los indicadores más simples y cuya definición no cambia, como por ejemplo **Ingreso Neto**. Estos indicadores son pocos comparados con los indicadores definidos por el usuario.
- **definidos por el usuario**: son indicadores que el usuario de alguna forma ingresará en el sistema. Los nuevos indicadores definidos por el usuario pueden ser utilizados directamente o ser guardados para un uso posterior.

Se pide:

1. Diseñar, implementar e incorporar al modelo de dominio los indicadores, de forma que puedan ser tanto cargados por el analista de inversiones como soportados nativamente por la aplicación. En cualquier caso, debe ser fácil agregar nuevos indicadores, y debe ser posible evaluarlos contra cualquier empresa en cualquier año. Es importante tener en cuenta que:

Sugerencia: Una forma usual de permitir que el usuario ingrese cálculos matemáticos simples es mediante un lenguaje específico de dominio (DSL), como lo son por ejemplo las expresiones que podemos encontrar en una hoja de cálculo como Google Docs Spreadsheet, Apache OpenOffice Calc o Microsoft Excel.

Eso implica *parsear* strings. Si bien es algo que se puede hacer (con mucho esfuerzo a mano), existen herramientas que simplifican mucho esto, como por ejemplo JavaCC o Antlr. Recomendamos utilizar alguna de estas herramientas.

Advertencia: si se utiliza un DSL para introducir indicadores, es necesario contemplar que el usuario con seguridad ingresará fórmulas incorrectas ocasionalmente, ya sea por sintaxis inválida o por emplear cuentas u otros indicadores en ellas que no existan. El sistema debe manejar adecuadamente estos ingresos incorrectos.

2. Extender las vistas para poder soportar:
 - a. la carga de indicadores definidos por el usuario
 - b. el guardado y recuperación de indicadores definidos por el usuario
 - c. listar a los indicadores junto con las cuentas de una empresa en un cierto período.
3. Definir y Generar los casos de prueba para dar una cobertura adecuada a la presente entrega.

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Desarrollo

Gramática

La gramática que definimos para poder diferenciar indicadores de cuentas es la siguiente:

Indicadores

Comienzan con i_ siempre y en minúscula, luego puede seguir con cualquier combinación de letras mayúsculas, minúsculas y guiones bajos.

Cuentas

Comienzan con c_ siempre y en minúscula, luego puede seguir con cualquier combinación de letras mayúsculas, minúsculas y guiones bajos.

Diagrama de clases

Link: <https://drive.google.com/file/d/0B4BM6drJEYLT3V1UmlIX09qWFk/view?usp=sharing>

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

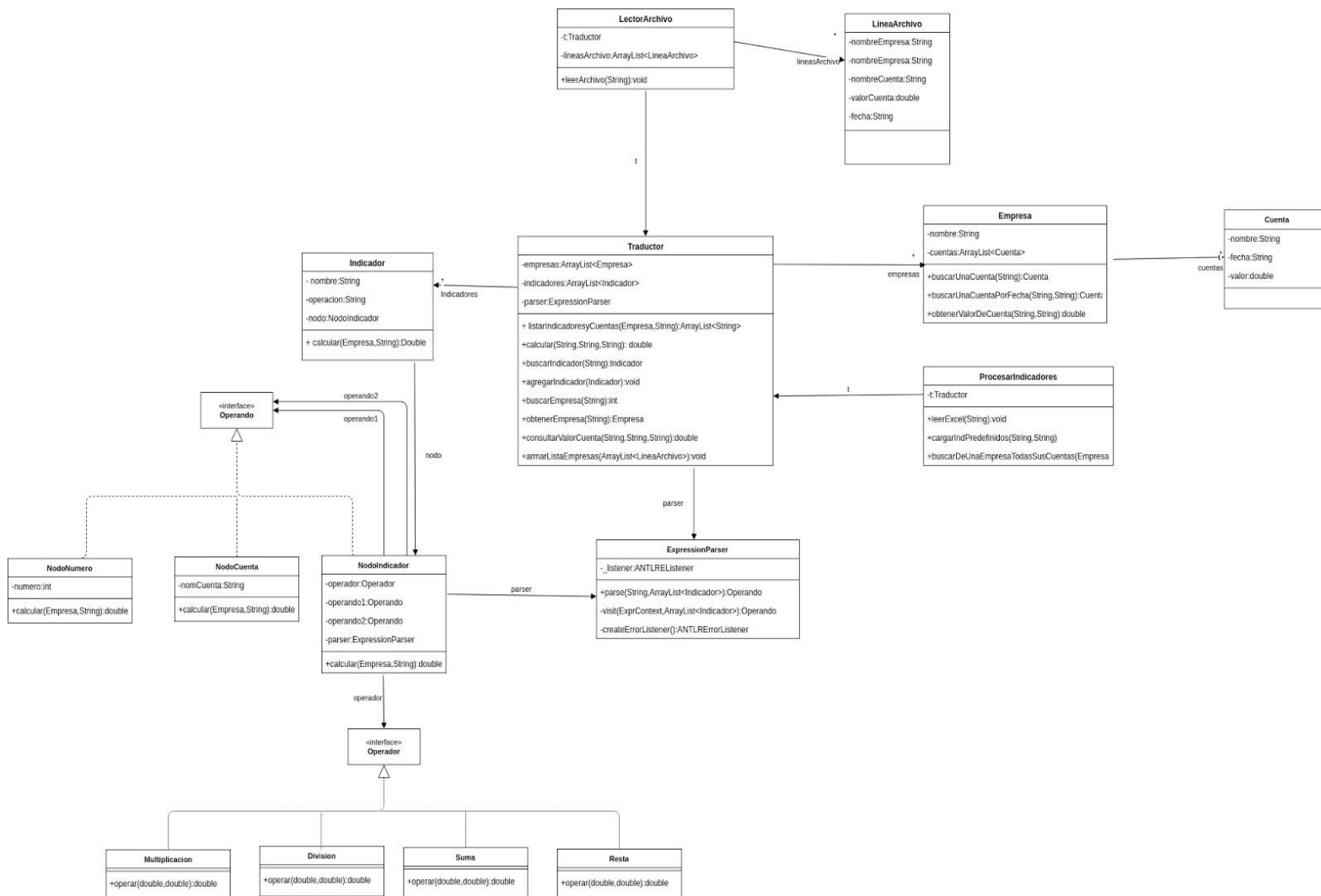
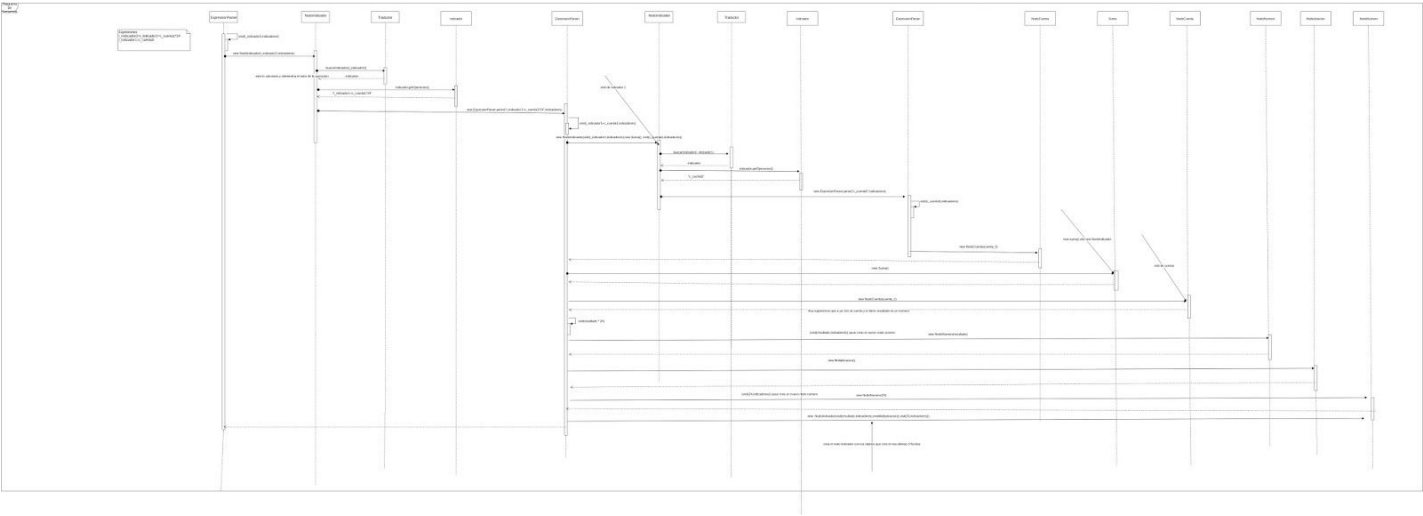


Diagrama de Secuencia

Link: <https://drive.google.com/a/frba.utn.edu.ar/file/d/0BwhF2EUYISFuOGFxbUxJUHR3ak0/view?usp=sharing>

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0



Entrega 3

Enunciado

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0



Tercera Entrega: Metodologías

Llegó finalmente el momento de implementar las metodologías, así que el plato fuerte de esta iteración es su diseño y desarrollo.

Se pide:

1. Diseñar e implementar las condiciones y metodologías, de forma que las 4 primeras condiciones de metodología de Buffet puedan ser evaluadas, y que sea fácil definir metodologías personalizadas similares.

Sugerencia: A la hora de diseñar las condiciones, prestar atención a que:

- Hay algunas que son taxativas (*en esta empresa se puede invertir, en esta no*), otras que sirven para priorizar (*en esta empresa conviene invertir más que en otra*), y algunas que combinan ambas ideas
- Revisar las condiciones de ejemplo e identificar cuáles son primitivas y cuales son derivadas de las anteriores. Esto te ayudará a la hora de modelarlas.

Advertencia: Releer con cuidado la especificación de las metodologías. Puede que haya aspectos no definidos o ambiguos que tengas que discutir con tu docente.

2. Diseñar e implementar un mecanismo para que el analista pueda definir sus propias metodologías desde la interfaz gráfica.

Advertencia: No es posible implementar una funcionalidad que le permita al analista definir cualquier tipo de metodología, dado que siempre pueden surgir cierto tipo de análisis complejos no contemplados por la herramienta que estamos diseñando. Parte de esta entrega es encontrar, con ayuda de tu docente, una solución de compromiso entre contar con un alta de metodologías flexible pero realizable en el tiempo del trabajo práctico.

3. Diseñar e implementar las pantallas necesarias para que un analista seleccione una metodología (ya sea predefinida o cargada por el usuario) y la ejecute contra todas las empresas cargadas en el sistema. El resultado debe ser un listado ordenado de empresas en las que es deseable invertir

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Desarrollo

Para iniciar la aplicación desktop es necesario dirigirse a la clase main del paquete ubicado en main/src/interfazGrafica.

Una vez iniciada la aplicación aparecerá un menú en el cual aparecen 3 opciones:

Cargar Empresas y cuentas

Para probar todas las funcionalidades es necesario ingresar a este menú y escribir en el campo cargar archivo Datos.txt, esto cargará el archivo cuentas y empresas al hacer clic en el botón cargar.

Una vez hecho esto se puede consultar el valor de una determinada cuenta de una empresa en un periodo, completando todos los campos y luego clicar el botón obtener.

Cargar Indicadores

Para probar todas las funcionalidades es necesario ingresar a este menú y escribir en el campo cargar archivo Indicadores.xls, esto cargará el archivo indicadores al hacer clic en el botón cargar.

Una vez cargado este archivo se puede calcular el indicador a una empresa determinada en determinado periodo.

Además se pueden crear nuevos indicadores, es importante a la hora de ingresar el nombre del indicador a ingresar ingresar con i_ como prefijo, y seguido el nombre del indicador.

Menú Metodologías

Una vez cargados los dos archivos(Datos.txt e Indicadores.xls), esta interfaz permite crear una metodología, agregar condiciones a la metodología cargada y aplicar la metodología a todas la empresas presentes en el archivo Datos.txt.

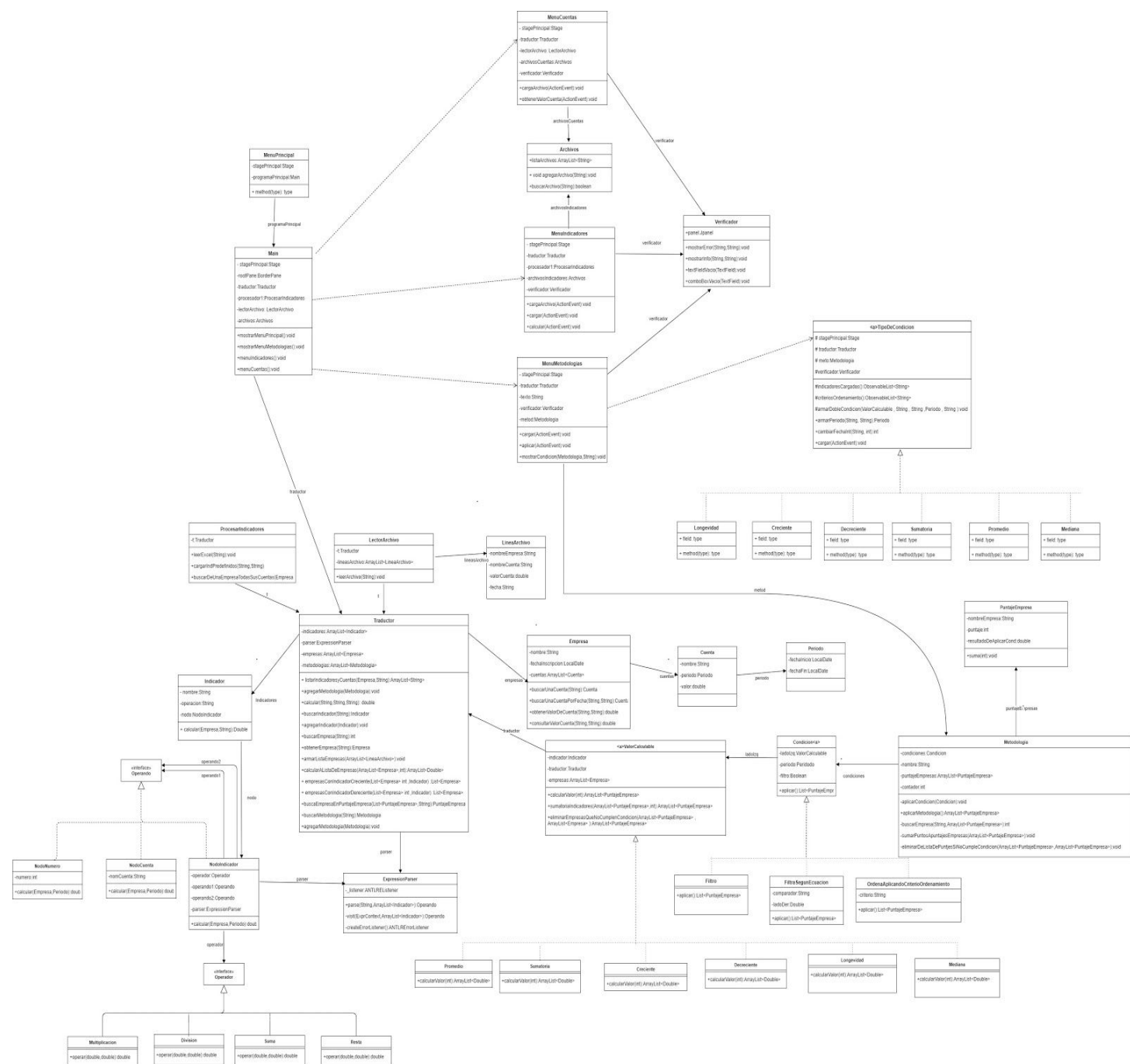
Cabe resaltar que las condiciones de una metodología son de 6 tipos :

Promedio,Sumatoria,Mediana,Creciente,Decreciente y Longevidad cada una de estas condiciones cuenta con su propia interfaz a la hora de cargar una condición a la metodología. Además dependiendo el tipo de condición:

- se filtra empresas, es decir, se eliminan empresas que no cumplen la condición.
- se filtran empresas y las que quedan se ordenan según un criterio definido por el usuario,según como este considera que debe ordenarse el puntaje obtenido por la empresa al aplicar la condición en cuestión.

Diagrama de clases

Link: <https://drive.google.com/file/d/0B4BM6drJEYLJT3V1UmlIX09qWFk/view?usp=sharing>



<i>Diseño de Sistemas</i>	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Entrega 4

Enunciado

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Dónde invierto?	Grupo: 12 – Versión 1.0

¿Dónde invierto?

Segunda Parte (Segundo Cuatrimestre)

1 Contexto

A la vuelta de las vacaciones de Sabrina, Héctor le cuenta que en estas cuatro semanas las cosas cambiaron mucho.

Primero, porque ahora el mismo Héctor tiene serias dudas sobre su idea original: llegó a la conclusión de que nada es tan fácil como parece, que el mundo de la bolsa es demasiado competitivo y aún hay aspectos que no domina como para poder dar asesoramiento a más personas.

Segundo, porque charlando con un amigo, éste opinó que de todas formas el software que habían creado junto a Sabrina era una herramienta interesante para pequeños inversores, y que él consideraba que muchos estarían dispuestos a pagar por éste.

Todo esto lo llevó a Héctor a plantear un nuevo modelo de negocio para su incipiente emprendimiento: en lugar de asesorar a clientes, proveer directamente un software para quienes sí lo hacen. Y tras varias reuniones más llegó a Valeria, empresaria que está dispuesta a ayudar económicamente a desarrollar este nuevo producto.

Por teléfono, Héctor le acaba de preguntar a Sabrina si ella quiere ser la arquitecta de un nuevo sistema, esta vez Web y con soporte para múltiples usuarios, basándose en el anterior. Y tras pensarlo un poco, no sólo Sabrina dice que está dispuesta a hacerlo, sino que además, considera que puede reutilizar mucho del viejo sistema.

2 Entregas

2.1 Cuarta Entrega - Persistencia

El objetivo de esta entrega consiste en adaptar esta arquitectura para centralizar el almacenamiento. Es decir, necesitamos permitir almacenar los elementos del dominio en un medio relacional, no siendo necesario mantener la persistencia en archivos. La única excepción serán las cuentas, que por ahora seguiremos cargando tando desde la base de datos como desde un archivo.

Además, no queremos acoplarnos a una base de datos particular, y para poder reutilizar el modelo de objetos ya construido, se pide el uso de un ORM. Si bien parece fácil, ¡cuidado!, puede que haya que realizar modificaciones al modelo original. Si bien el ORM permite (en general) el fácil cambio del motor de base de datos, en este curso es obligatorio que tenga el driver y este configurado para el motor MySQL

Sugerencia: De utilizar Java 8 o posterior, recomendamos utilizar JPA/Hibernate (es decir, Hibernate a través de la interfaz estándar de JPA). Un buen punto de partida para adaptar el proyecto es el siguiente: <https://github.com/dds-utn/jpa-proof-of-concept-template>.

Se pide:

1. Introducir en la solución un motor de base de datos relacional
2. Introducir en la solución un motor ORM
3. Persistir en dicha base de datos las siguientes entidades:
 - a. Cuentas
 - b. Indicadores
 - c. Metodologías
 - d. Empresas
4. Lograr que la aplicación siga funcionando sin cambios funcionales
5. Lograr que la aplicación pueda seguir cargando cuentas desde archivos. No es necesario desarrollar nuevas pantallas, sino sólo continuar soportando esta carga programáticamente.
6. Se debe agregar a la documentación
 - a. Como configurar la base de datos y como cargar datos de prueba en ella.
 - b. El modelo físico elegido
 - c. La justificación de los mapeos escogidos

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Desarrollo

Para que la aplicación funcione de manera correcta, y se persistan los datos correctamente, se deberá crear un usuario “dds” con una contraseña “dds” mediante el uso de la consola de mysql server, con los siguientes comandos:

```
CREATE USER 'dds'@'localhost' IDENTIFIED BY 'dds';
GRANT ALL PRIVILEGES ON * . * TO 'dds'@'localhost';
FLUSH PRIVILEGES;
```

La creación de un nuevo esquema no será necesaria, la aplicación lo hace automáticamente.

La carga de los datos de prueba en la base de datos se hace mediante el uso de la aplicación como en la entrega 3. Al ingresar un archivo con las empresas y las cuentas, estas quedarán persistidas. Lo mismo sucederá con las metodologías y los indicadores, pero estos también se podrán crear de a uno, y en el caso de las metodologías, estas no podrán ingresarse con archivos.

Para esta entrega consideramos que sería útil que los archivos “Datos.txt” e “Indicadores.xls” se carguen automáticamente una vez iniciada la aplicación, de esta forma, quedarán persistidos sus datos “a la primera”.

Por último, en lo que respecta a las dos herencias persistidas, decidimos que lo más conveniente era el uso de una única tabla que tendría campos en nulo o no dependiendo del tipo de clase que era. Hicimos esto porque esas dos herencias compartían muchos atributos con sus respectivas clases hijas, y el uso de una única tabla (en cada caso), nos evitaría el uso de muchos JOIN, haciendo las consultas más performantes.

Aviso: Antes de correr los testeos de persistencia con JUnit, se deberá probar haciendo mvntest o mediante la aplicación para la creación automática del schema.

DER

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

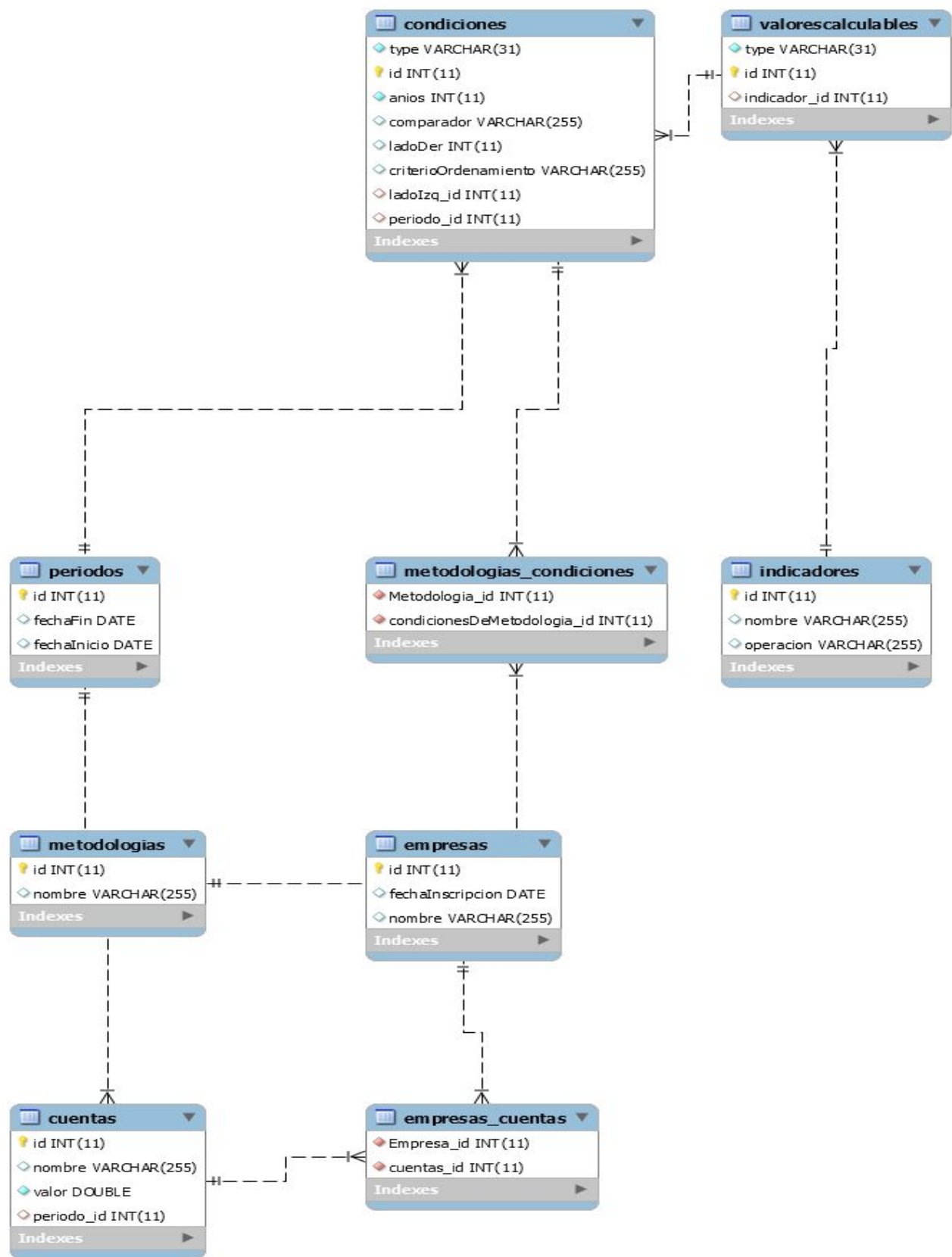
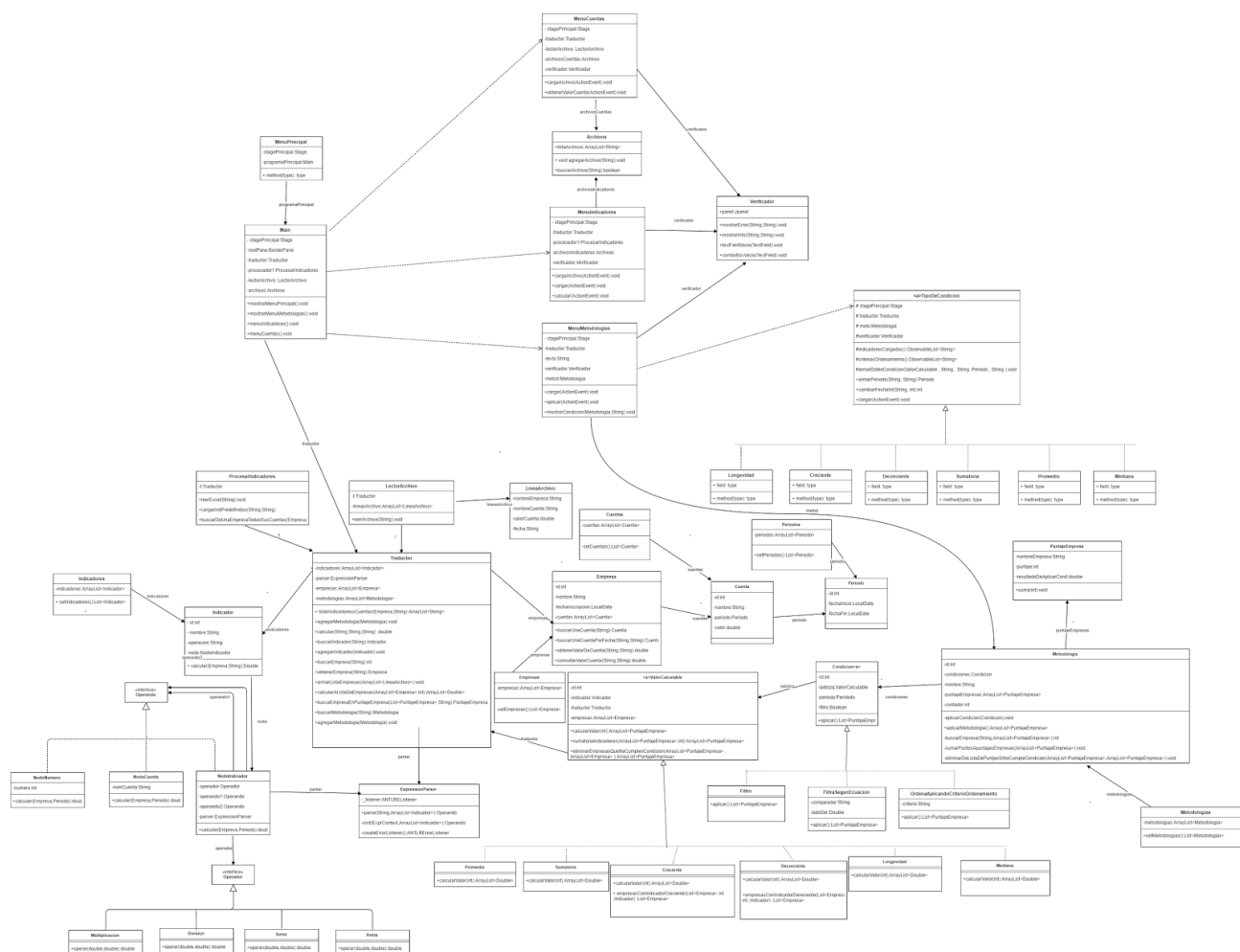


Diagrama de Clases



Link: <https://drive.google.com/file/d/0B4BM6drJEYLT3V1UmlIX09qWFk/view?usp=sharing>

<i>Diseño de Sistemas</i>	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Entrega 5

Enunciado

Quinta Entrega - UI Web

¡Llegó el turno de implementar las nuevas vistas! Para esta entrega, nuestro requerimiento es simple: necesitaremos implementar las pantallas Web necesarias para todas las funcionalidades existentes:

- Visualización de cuentas
- Creación de indicadores
- Evaluación de indicadores
- Creación de metodologías
- Evaluación de metodologías

Sugerencia: Spark (<http://sparkjava.com/>) es un framework muy simple para desarrollar aplicaciones Web en Java, que si bien no presenta funcionalidades avanzadas, resulta suficiente para esta entrega.

Por último, hay que tener en cuenta que ahora que el almacenamiento estará centralizado y la aplicación será accedida por múltiples analistas, las metodologías e indicadores serán propias de cada usuario. Por ejemplo:

- si Hernán crea un indicador A y B;
- y Anabel crea indicadores C y D;

Entonces Hernán sólo verá los indicadores A y B, y Anabel, C y D.

Sugerencia: En el marco de la presentación Web, para construir componentes visuales consistentes es necesario un grado considerable de experiencia en maquetado con HTML y CSS que excede los tiempos acotados del TP Anual. Por eso, recomendamos ampliamente el uso de frameworks como Bootstrap (<http://getbootstrap.com/>), que proveen componentes visuales y sistemas disposición de componentes que cubren los requerimientos planteados.

Se pide:

1. Reimplementar las vistas empleando una interfaz Web. No es necesario mantener compatibilidad con las interfaces de escritorio originales, que se podrán descartar.
2. Realizar los cambios arquitecturales necesarios para que la aplicación pueda ser servida a través de HTTP
3. Incorporar a la aplicación el concepto de login. **No** es necesario implementar:
 - a. un flujo de registración, sino que bastará con que sea posible cargar los usuarios a través de la base de datos;
 - b. una política segura de control de contraseñas
 - c. login con redes sociales

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

4. Describir mediante diagramas y prosa la arquitectura física y lógica del nuevo sistema, haciendo foco en los siguientes aspectos:

- Nodos de red desplegados
- Motores de persistencia
- Interfaces con sistemas internos y externos
- Protocolos de red utilizados
- Principales componentes lógicos de alto nivel

Indicar además qué acciones son necesarias para escalar horizontalmente a la nueva arquitectura.

Anexo curso martes noche (todo lo anterior es la entrega oficial de la materia, lo que está a continuación es un agregado de nuestros cursos):

- Los diagramas de la arquitectura física deben hacerse utilizando el diagrama de despliegue UML
- La estructura de las urls de la aplicación debe respetar las convenciones REST (no es solo para APIs). Por ejemplo para listar los indicadores, se debe hacer GET /indicador[es], para dar de alta uno se debe hacer POST /indicador[es] con los datos correspondientes, etc.
- Si aún no se aprobó la UI desktop, en alguna versión/branch tiene que estar funcionando, el hecho de que se descarte NO quiere decir que si hubo correcciones no se respeten.

Consideraciones

*En la base de datos hay cargados 4 usuarios para loguearse, y todo tienen en común solo 2 indicadores “i_NivelDeuda” e “i_MargenVentas”. Luego por cada indicador que crea el usuario solo este usuario podrá usarlo. Los usuarios para loguearse son:

Nombre	Password
Lucas	123
Nicolas	123
Gabriel	123
Daniel	123

-Antes de calcular indicadores o metodologías hay que ir al menú obtener empresas para que se carguen las

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

empresas.

Arquitectura

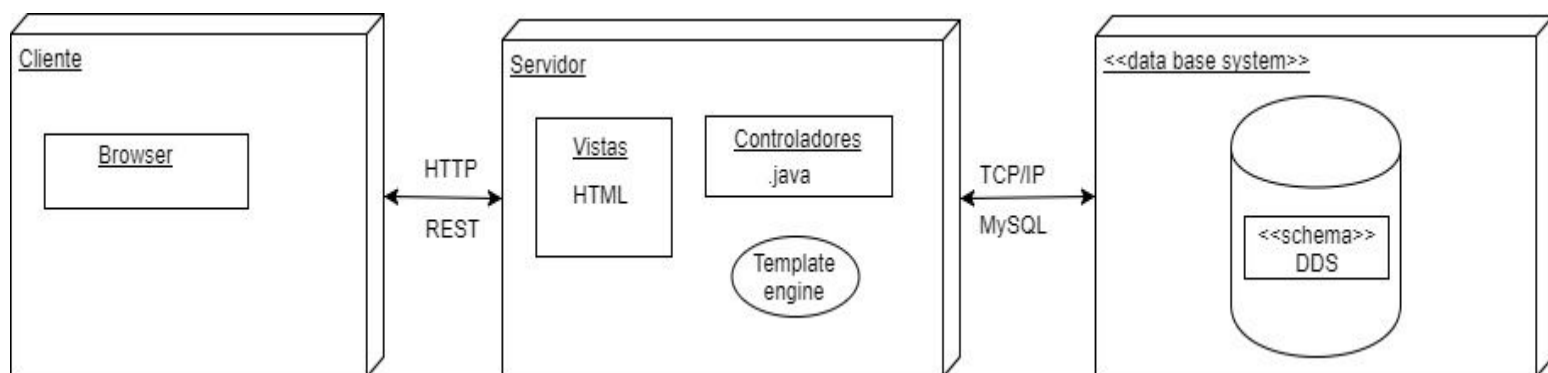
El proyecto consta de una aplicación web basada en la arquitectura cliente-servidor, donde el usuario puede realizar diversas acciones mediante el uso del servicio REST y mediante el protocolo HTTP. Toda consulta que realice el usuario son atendidas por el servidor, quien tiene una conexión con una base de datos centralizada en un único nodo, mediante el uso del motor MySQL.

El lenguaje utilizado es Java, es orientado a objetos, por lo que la persistencia de los datos se realiza con una implementación de Java persistence API (JPA) llamada Hibernate.

Otro enfoque de la arquitectura es el estilo Modelo-Vista-Controlador (MVC) separando al modelo de la aplicación, la información que ve el cliente y el controlador (que funciona como un intermediario entre el modelo y la vista).

Si en un futuro se quisiera escalar horizontalmente se podría descentralizar la base de datos y ubicarla en distintos nodos. También se tendrían más servidores que atiendan consultas para evitar el único punto de falla o “cuello de botella” y un balanceador de carga que los coordine. De esta forma se descartaría el paradigma puramente relacional y se haría uso de NoSQL, agregando nodos cada vez que sea necesario escalar de forma horizontal

Diagrama de despliegue



Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Diagrama de componentes

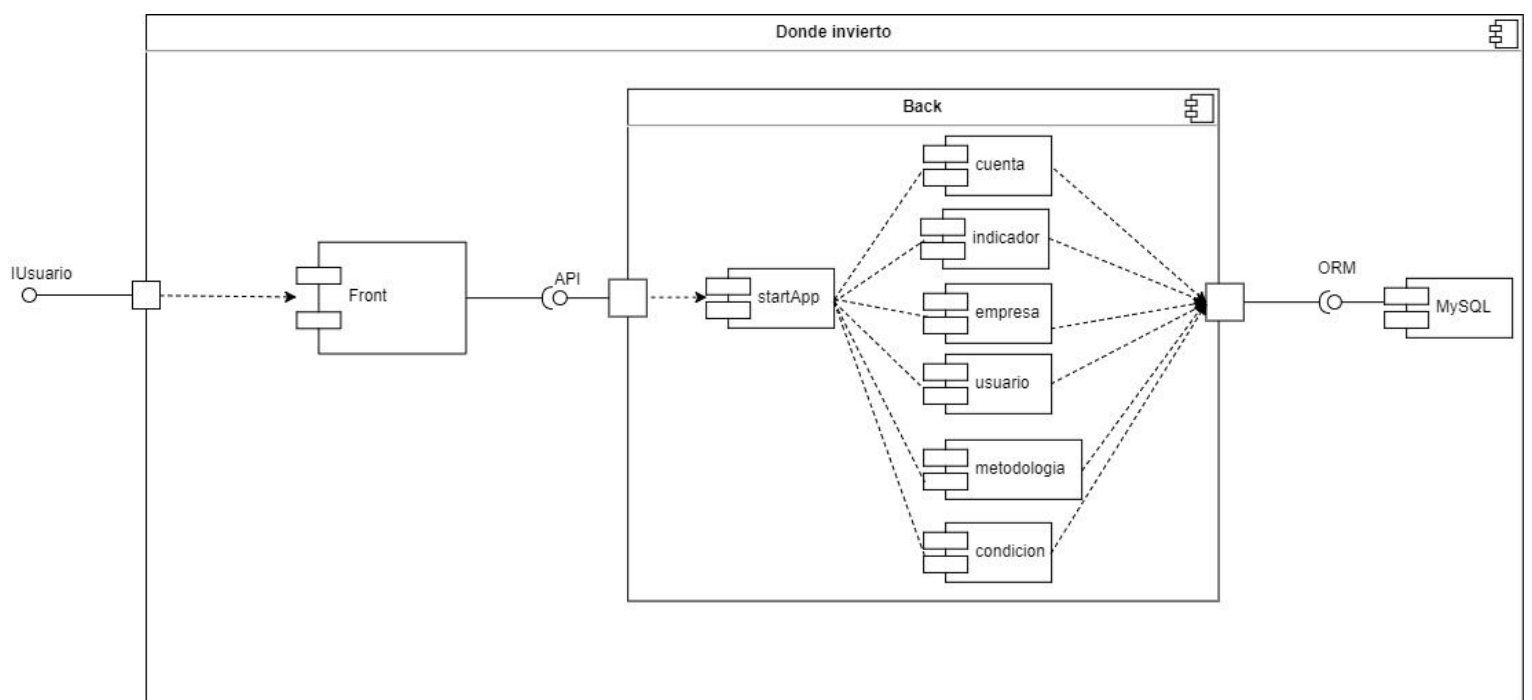
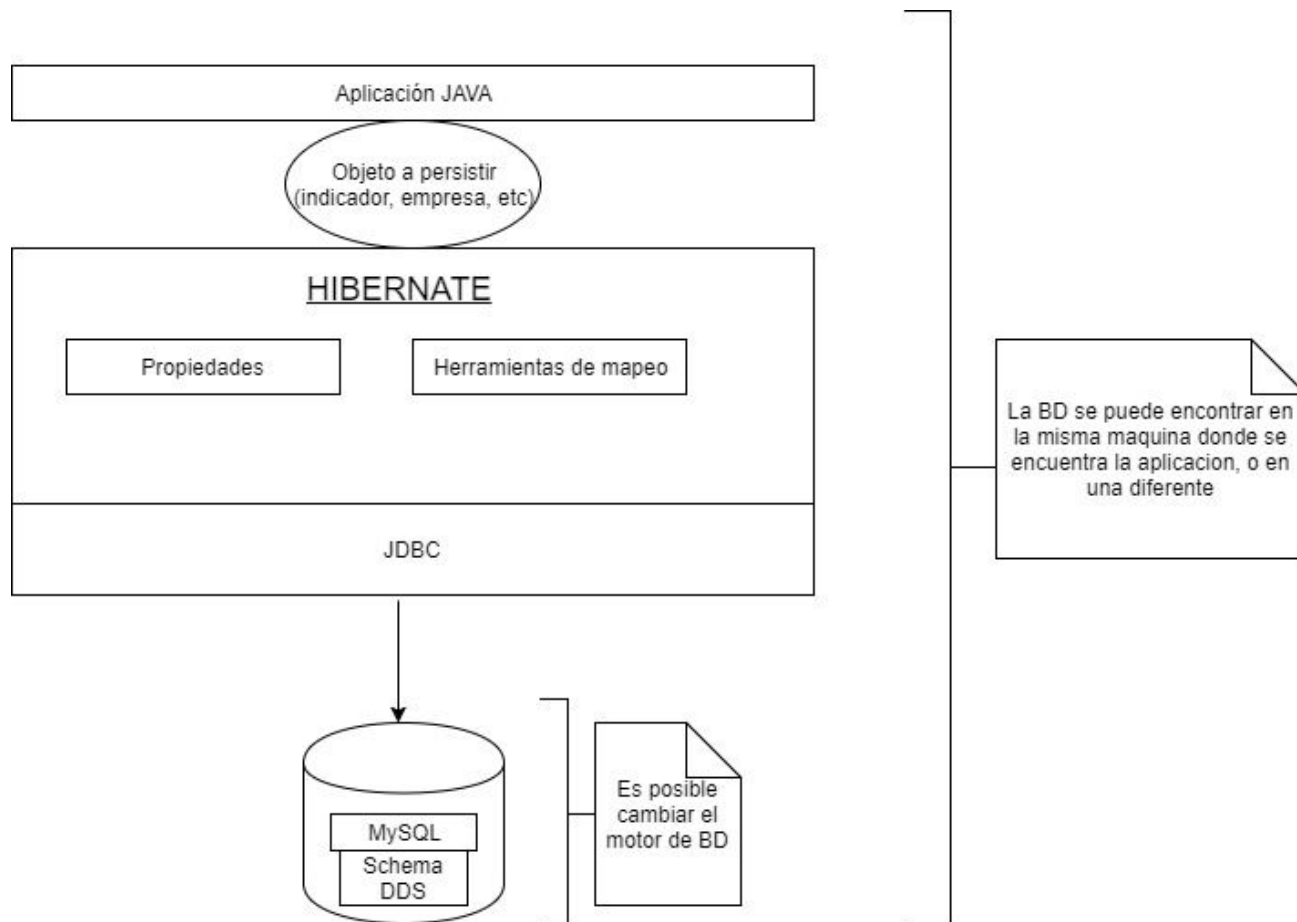


Diagrama de persistencia

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0



Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Diagrama de protocolos de Red

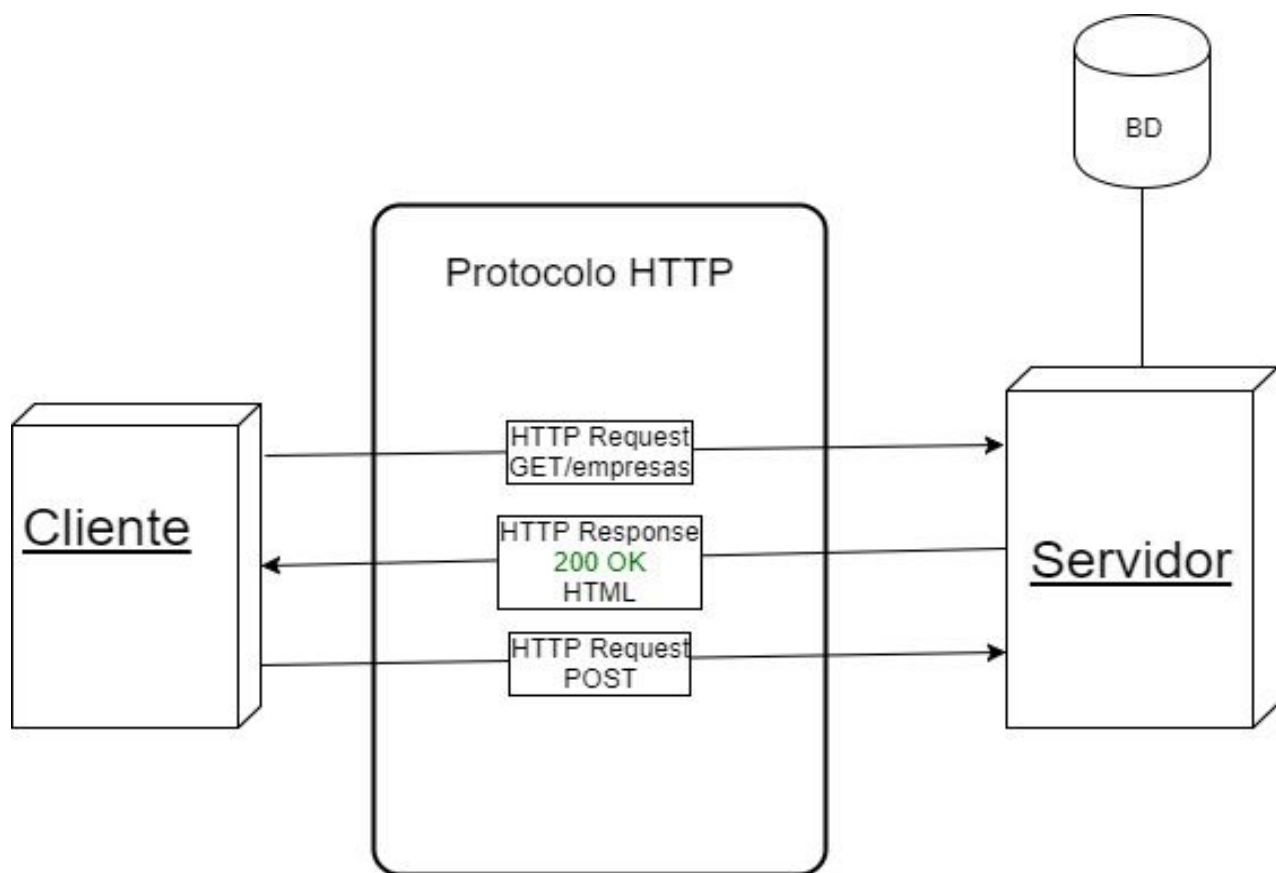


Diagrama de interfaces con sistemas internos y externos

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

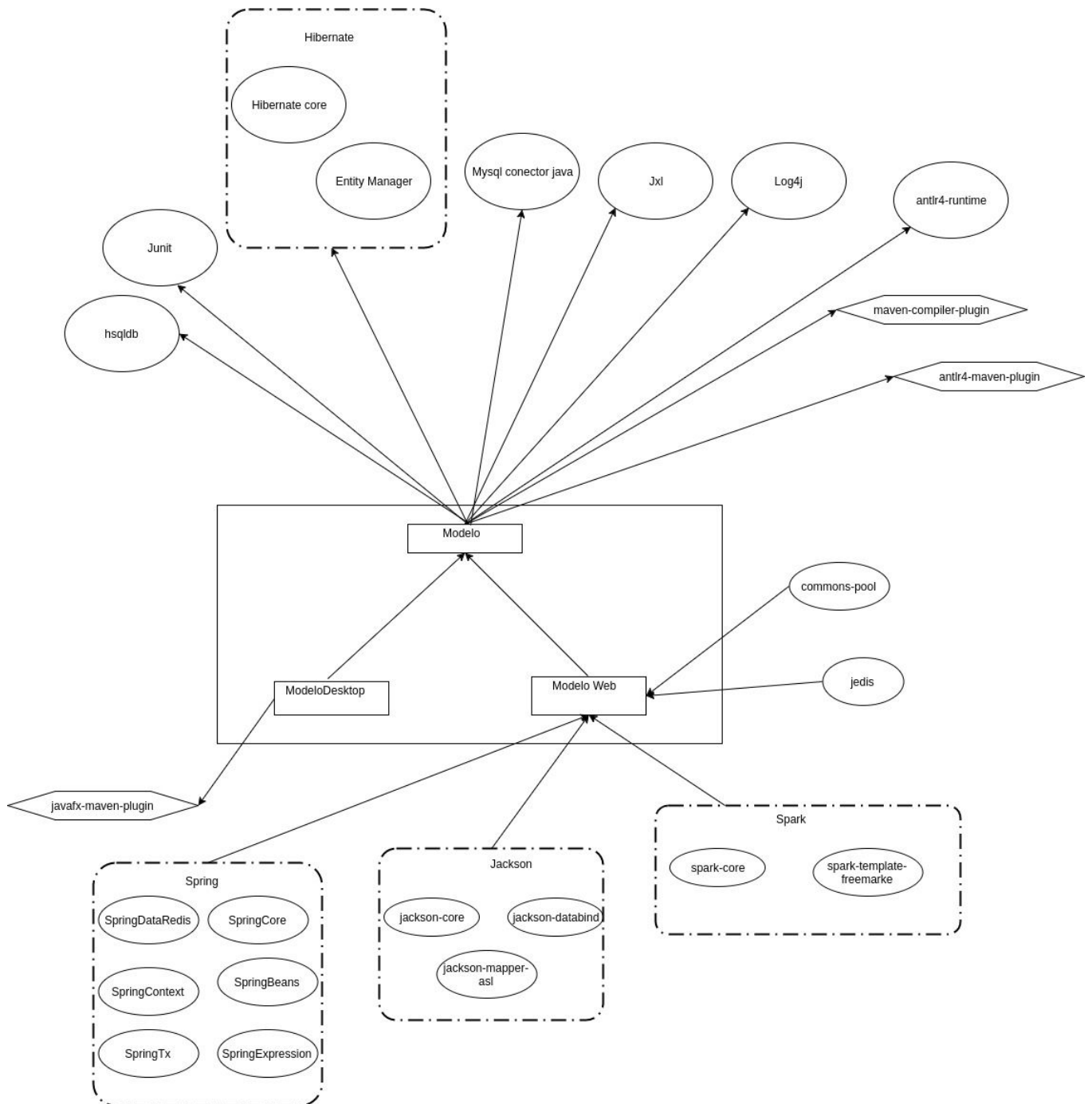


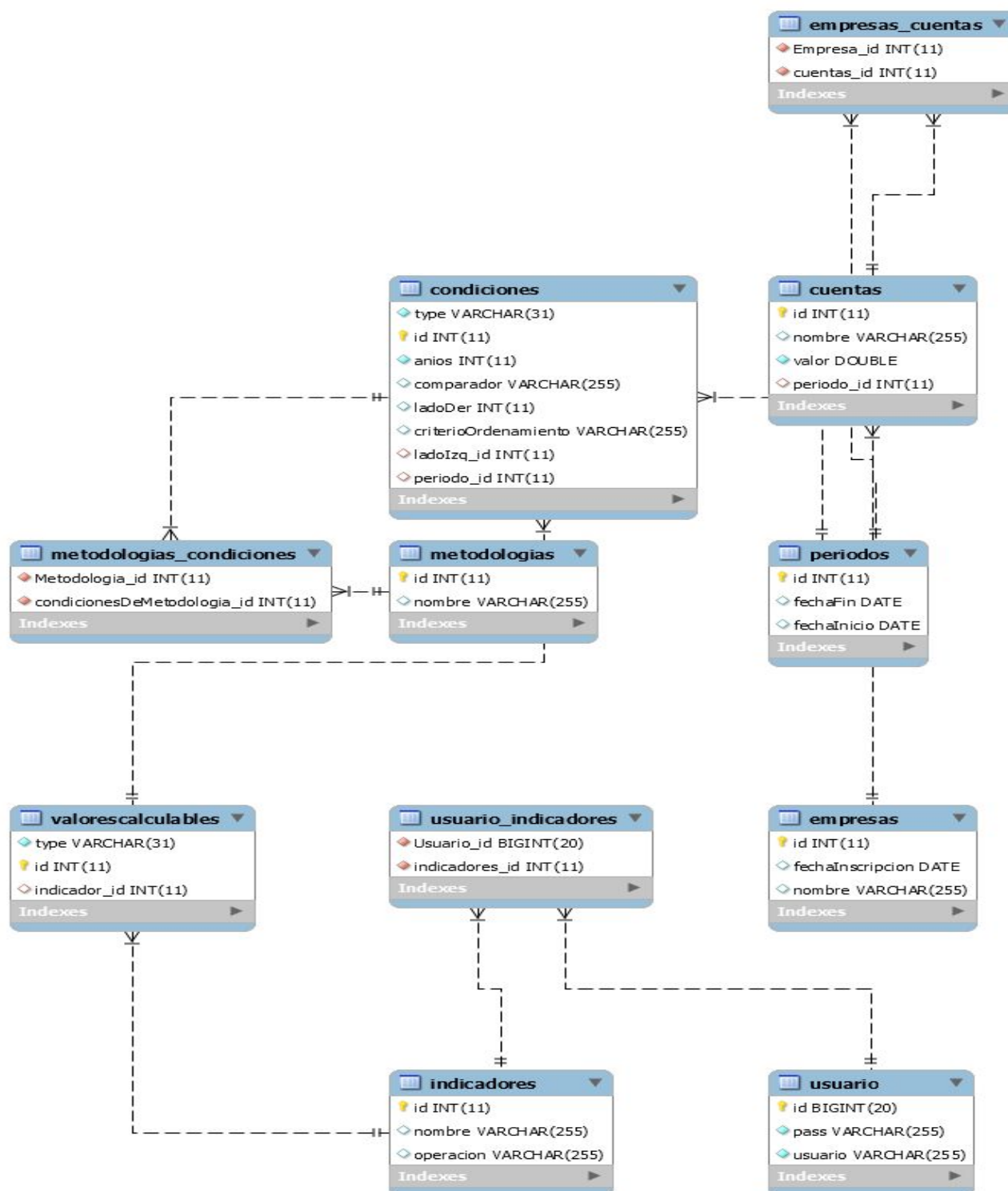
Diagrama de clases

Para verlo correctamente ir al link y seleccionar abrir con draw.io Diagrams



Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Dónde invierto?	Grupo: 12 – Versión 1.0

Diagrama de entidad de relación



Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Entrega 6

Enunciado

1. Introducir múltiples fuentes de datos provenientes de internet.
2. Introducir un proceso de carga de datos offline.
3. Desnormalizar el cálculo de ratios en una base Mongo o alguna otra base de datos no-relacional
4. Deploy. Indicar además qué acciones son necesarias para escalar horizontalmente a la nueva arquitectura.

Anexo curso martes noche

Todo lo anterior es la entrega oficial de la materia, lo que está a continuación es una modificación para nuestros cursos.

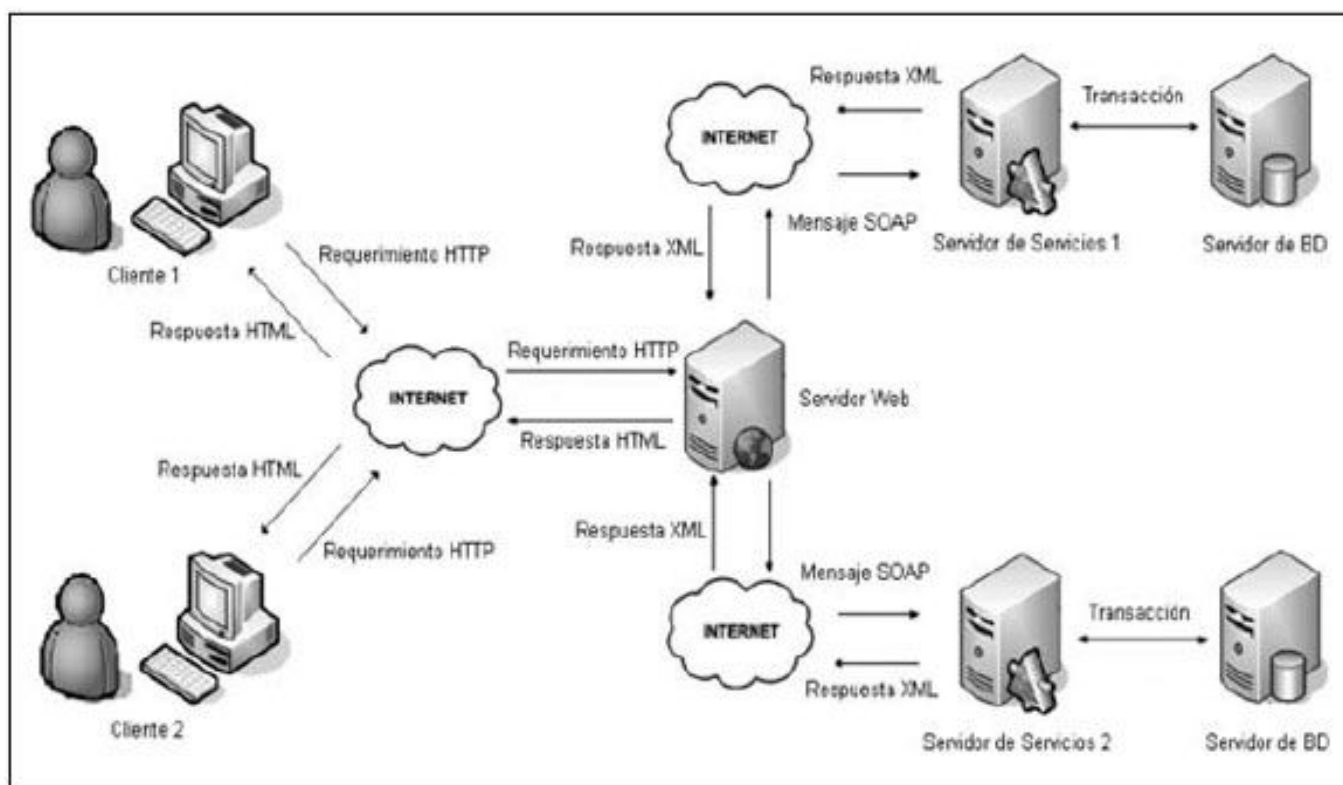
- Los puntos 1, 2 y 4 no deben implementarse, pero si se debe realizar un diseño y agregarlo a la documentación. Se espera que se transmita el detalle suficiente para comprender la solución propuesta.
- En particular el punto 4 requiere un poco de investigación por parte del alumno, no lo dejen para ultimo momento.
- El objetivo del punto 3 es mejorar el rendimiento para los cálculos de los indicadores. Puede optar por:
 - Hacer una cache a nivel aplicacion sobre una base clave/valor (Redis por ejemplo)
 - Realizar una carga de datos en una base documental con todas las combinaciones de indicador/cuenta posibles de una empresa dada y corroborarlo mediante tests.

RECUERDEN QUE DEBE QUEDAR UN ÚNICO PDF CON TODA LA DOCUMENTACIÓN SOLICITADA DE TODO EL AÑO. ES CONDICIÓN NECESARIA PARA RENDIR LA ENTREGA FINAL.

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

Punto 1

Múltiples Fuentes de datos online:



Para este caso lo que se planteó es el concepto de Web Services donde estamos con la necesidad de intercambios de datos entre múltiples fuentes de forma online. El término Web Services describe una **forma estandarizada de integrar** aplicaciones WEB mediante el uso de XML, SOAP, etc.. sobre los protocolos de Internet. XML es usado para describir los datos permitiendo habilitar definiciones, transmisiones, validaciones, e interpretación de los datos entre aplicaciones y entre organizaciones, y SOAP se ocupa para la transferencia de los datos, este último es independiente del sistema operativo y puede ser transportado por los distintos protocolos como es el caso de HTTP.

Para nuestro caso las múltiples fuentes serían consultas a diferentes páginas web que contengan información de empresas y cuentas necesarias, mientras que el browser, el cliente, de la pc del usuario de nuestra aplicación WEB es el que necesita de esta información y por esto el programa requerirá consultar de cada web services correspondiente a cada una de las fuentes de datos.

Los Web Services **permiten a las organizaciones intercambiar datos** sin necesidad de conocer los detalles de

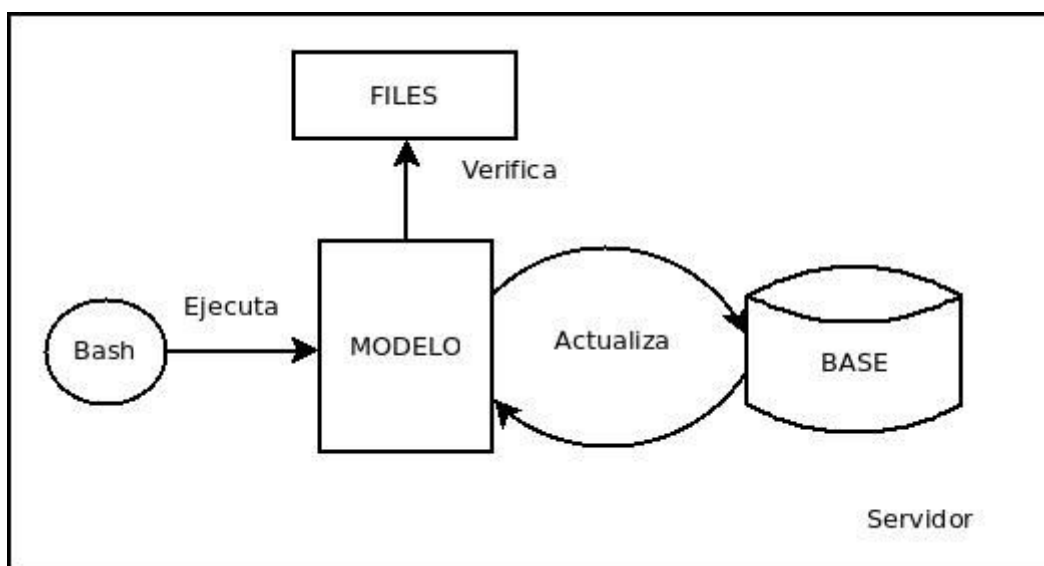
Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

sus respectivos Sistemas de Información, este es un punto fundamental ya que nuestra aplicación WEB no necesita conocer en detalle las fuentes, sino que lo único que importa es generar un web services compartiendo la lógica del negocio, los datos y los procesos, por medio de una interfaz de programas a través de la red.

En nuestro caso particular para la implementación creamos una clase que consulta por un servicio a una url específica, los datos con los que nos responde son de tipo json, entonces esta clase mapea los datos respuesta de ese servicio a un objeto de CuentasJson, a un dato json lo convierte en un objeto cuenta java (mediante annotations), y retorna una lista de CuentasJson.

Además se podría usar un objeto DTO (Data Transfer Object) que permite optimizar recursos. El objetivo de este objeto es realizar menos consultas a un web service ,ya que estas consultas son costosas, por ello es que el objeto se trae todos los datos que necesita de una vez.

Punto 2



En la resolución de este punto se planteó el concepto de bash para introducir de alguna forma el concepto de carga de datos offline. En principio el escenario que se planteó es que se producen incorporaciones masivas de datos offline diariamente cuando la aplicación no se encuentre en uso online, evitando inconsistencia de datos ya que al momento de actualizar no estaría en uso la aplicación.

Este lenguaje de Shell nos permitiría trabajar por tiempo o por suceso, se utiliza habitualmente como administrador de sistemas operativos y particularmente puede realizar acceso a la base de datos, como también tiene el poder de verificar si un servicio/proceso se está ejecutando. Lograríamos de esta manera que se ejecute periódicamente verificando si nuestro servicio/proceso se está ejecutando, cuando no se esté ejecutando ordenaría que se envíe la carga masiva de los datos offline hasta terminar o incluso en caso de ser un gran volumen de datos y necesite un tiempo prudencial, verificar cada tanto si el servicio/proceso

Diseño de Sistemas	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0

ha iniciado nuevamente para evitar inconsistencias de datos. También se podría incluir un bash para que no se ejecute el servicio/proceso hasta que no termine de realizar la carga periódica offline.

Tener en cuenta que a nivel usuario se pierde el control sobre cómo es el proceso y retoma el control una vez finalizado el proceso.

Una manera de realizar los tests de esta implementación, es utilizar el Files correspondientes a los datos offlines que se quieren ingresar a través del modelo si se persistieron en la base de datos, realizar consultas en los tests en las distintas entidades (metodologías, indicadores, cuentas, empresas) y verificar que se haya implementado. También se puede tener un archivo sysout.log que informe si el bash se está corriendo periódicamente y que informe la cantidad de procesos que se realizaron en cada corrida.

Punto 3

Utilizamos Redis para la implementación de una memoria cache y así resolver este punto, por lo cual antes de iniciar la aplicación, se deberá iniciar redis-server desde la consola.

Los test hechos para este punto se encuentran en la carpeta “test” del modelo web.

Punto 4

Para escalar horizontalmente la arquitectura, la tecnología utilizada deberá ser NoSql haciendo que se deba estar dispuesto a sacrificar la consistencia o la disponibilidad de la base de datos (propiedad CAP). Además, se deberá incorporar varios nodos haciendo que estos trabajen como un “todo”, esta red de servidores es conocida como “Cluster”. Debe existir un servidor primario desde el cual se administra el cluster. Cada servidor del cluster deberá tener un software que permita integrarse al cluster, para las aplicaciones Java, tenemos los servidores de aplicaciones como Weblogic, Wildfly, Websphere, etc. y sobre estos se montan las aplicaciones que queremos escalar. Como el cluster es único, existe un único punto de falla, por eso se recomienda tener un cluster espejo para asegurar la disponibilidad del sistema.

Además, es importante que el servidor sea stateless o casi stateless, porque cuando el seguimiento de una aplicación depende del lado del servidor, entonces la sesión del usuario estará vinculada a ese servidor en particular. Si, por otro lado, todos los detalles relacionados con la sesión del usuario se almacenan en el lado del navegador, esa sesión se puede pasar sin problemas a través de literalmente cientos de servidores. La capacidad de entregar una sola sesión (o miles o millones de sesiones individuales) en los servidores de manera intercambiable es el verdadero paradigma de la escala horizontal.

La arquitectura debería estar orientada a servicios, es decir que deberá contener bloques lógicos, autónomos pero interactivos y así se podrá escalar cada uno de esos bloques de forma independiente a medida que su uso requiere la carga. La aplicación web deberá ser independiente de la aplicación en si y del almacenamiento cache.

<i>Diseño de Sistemas</i>	Curso: K-3051 – Año 2017
Trabajo Práctico: ¿Donde invierto?	Grupo: 12 – Versión 1.0