Documento de Arquitectura de Software

Grupo [X]

[Integrantes]

**Contenido**

[**1 Introducción. 3**](#_gjdgxs)

[**2 Vista de Casos de Uso 4**](#_30j0zll)

[2.1 Casos de uso identificados 4](#_1fob9te)

[2.2Diagrama de Casos de Uso Críticos 5](#_2et92p0)

[2.3 Actores 5](#_tyjcwt)

[*2.3.1Usuario 5*](#_3dy6vkm)

[*2.3.2 Administrador 5*](#_1t3h5sf)

[2.4 Especificación de Casos de Uso Críticos 6](#_4d34og8)

[**3 Vista Lógica 7**](#_2s8eyo1)

[3.1 Estilo Arquitectónico. 7](#_17dp8vu)

[7](#_3rdcrjn)

[3.2Subsistemas 8](#_26in1rg)

[*3.2.1Business Layer. 8*](#_lnxbz9)

[3.2.1.1 Diagrama de Componentes. 8](#_35nkun2)

[3.2.1.2Descripción. 8](#_1ksv4uv)

[3.3 Diagramas de Interacción 9](#_44sinio)

[**4 Vista de Distribución (Deployment). 10**](#_2jxsxqh)

[4.1Escenario 1 10](#_z337ya)

[*4.1.1 Descripción 10*](#_3j2qqm3)

[*4.1.2 Nodos 10*](#_1y810tw)

[*4.1.3 Conexiones 10*](#_4i7ojhp)

[**5 Vista de Implementación. 11**](#_2xcytpi)

[**6 Referencias 12**](#_1ci93xb)

# Introducción.

[En esta sección se realiza una introducción al sistema cuya arquitectura se describe. Para la elaboración de esta plantilla, se tomó como base el modelo 4+1 tal como fue presentado en el teórico del curso. El ejemplo que se brinda a continuación se basa en el laboratorio del año 2012 de TSI1.]

INDIGNADOFRAMEWORK es una plataforma informática de soporte a movimientos de protesta. Dicha plataforma tiene como finalidad centralizar toda la información relevante de un movimiento de protesta en un único portal web, el cual debe actuar como punto de integración de toda la información existente en la web asociada al evento. Esta información puede provenir de orígenes tan diversos como YouTube, Twitter, BBC, CNN, Wikipedia, etc. La plataforma también permite acceder a las últimas novedades, compartir recursos (imágenes, video, links), coordinar actividades, y facilitar la interacción entre los participantes de un mismo movimiento.

El sistema cuenta con dos módulos, uno de front office, de acceso público, con autenticación integrada con Facebook, y otro de back office para la administración del sistema, con autenticación restringida para un grupo reducido de administradores.

La plataforma brinda además soporte para la integración entre instalaciones distintas de INDIGNADOFRAMEWORK, permitiendo potenciar ciertas funcionalidades tales como ampliar la base de usuarios a los que les llegan las invitaciones a las convocatorias, búsqueda y chat entre usuarios en línea de distintas instalaciones.

# Vista de Casos de Uso

[Los elementos de las vistas desde las que se analiza la arquitectura son “ejercitados” por un pequeño subconjunto de casos de uso (o escenarios de esos casos de uso) que se denominan casos de uso críticos. Este subconjunto se debe elegir cuidadosamente, utilizando principalmente dos criterios:

* Que intervengan el mayor número de componentes arquitectónicos posibles
* Que intervengan los componentes que se vislumbran como críticos o más complejos.

A modo de ejemplo, se presentan los casos de uso críticos identificados por el Grupo 6 de la edición 2012 de TSI1 para la plataforma INDIGNADOFRAMEWORK.]

## ***Casos de uso identificados***

Módulo Front office:

 *Registro de Usuario.*

Registrarse como nuevo usuario en el portal del movimiento.

 *Autenticación de usuario.*

Los usuarios registrados tienen la posibilidad de loguearse para acceder a funcionalidades privilegiadas.

 *Crear convocatoria.*

Los usuarios registrados del sistema tienen la posibilidad de crear convocatorias en el movimiento de protesta. Notificándoles automáticamente a todos los usuarios que se encuentren en una locación cercana a la misma.

 *Compartir contenido.*

Los usuarios registrados podrán compartir información.

 *Chatear.*

Los usuarios podrán seleccionar un usuario que se encuentre en línea e iniciar una conversación por chat con el mismo.

 *Ver usuarios en línea.*

El usuario podrá ver los usuarios que se encuentren logueados en ese momento en el sistema.

Módulo Back office:

 *Crear movimiento.*

Consiste en dar de alta un nuevo movimiento a partir de templates y opciones de configuración provistas por el sistema.

 *Agregar fuente de noticias externa.*

El administrador podrá instalaren el sistema una fuente de noticia externa a partir de un plugin, para luego incorporarla en portales existentes.

 *Crear galería de imágenes.*

El administrador podrá crear una galería de imágenes para el portal del movimiento, tanto a la hora de crearlo o posteriormente.

 *Editar estilo.*

El administrador cuenta con la posibilidad de editar el estilo visual de un movimiento de protesta.

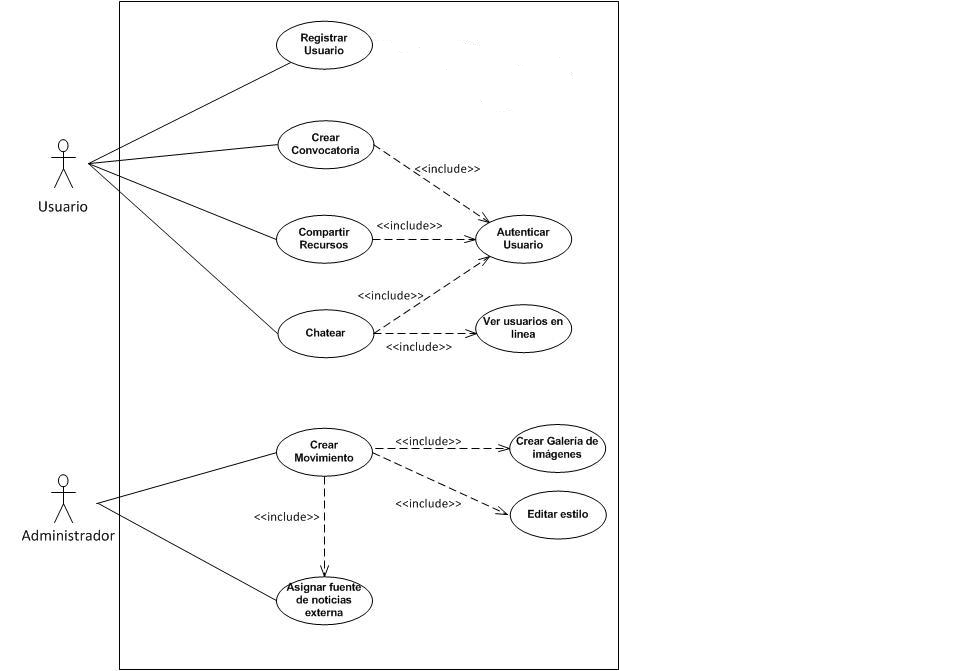
## 

## Diagrama de Casos de Uso Críticos

[En esta sección se utilizan los Diagramas de Casos de Uso de UML para mostrar los casos de uso críticos para la arquitectura. Los elementos a utilizar en estos diagramas son:

| Caso de Uso | Secuencia de acciones con un fin específico (Ej. Enroll in Unversity) |
| --- | --- |
| Actor | Persona, organización o sistema externo que interactúa con el sistema (Ej. Student) |
| Asociación | Relación entre un Caso de Uso y un Actor que participa en él. |
| Frontera del Sistema | La Frontera del Sistema se indica con un rectángulo que encierra los Casos de Uso. |

]



## Actores

[En esta sección se realiza una descripción de cada actor perteneciente al diagrama anterior.]

### **Usuario**

Personas que están registradas en algún movimiento de IndignadoFramework.

### **Administrador**

Son quienes crean, configuran y monitorean los diferentes movimientos de IndignadoFramework.

## Especificación de Casos de Uso Críticos

[En esta sección se incluye un ítem por cada caso de uso mostrado en el diagrama anterior. De cada caso de uso debe brindarse su descripción, así como uno o más flujos de eventos que detallen la interacción actor-sistema para cada escenario relevante del caso de uso. Opcionalmente, pueden especificarse pre y postcondiciones para cada caso. Las precondiciones son las condiciones que deben cumplirse para que el flujo del caso de uso pueda realizarse. Las postcondiciones son las condiciones que se cumplen al terminar el flujo del caso de uso (modificaciones en el estado del sistema).]

| **Nombre**: Autenticar Usuario |
| --- |
| **Objetivo**: Autenticar al Usuario en el portal web. |
| **Actores**: Usuario. |
| **Precondiciones**: - |
| **Descripción:** El Usuario ingresa su nombre y password, el sistema valida los datos e ingresa al usuario al portal web. |
| **Flujo Normal**:  1. El Usuario ingresa su nombrey password.  2. El Sistema verifica que los datos ingresados sean correctos. |
| **Flujos Alternativos**:  2A. La combinación nombre de usuario/password es incorrecta.   1. El sistema despliega un mensaje de error acorde. 2. Vuelve a 1. |
| **Poscondiciones**: El usuario queda logueado en el sistema. |

# Vista Lógica

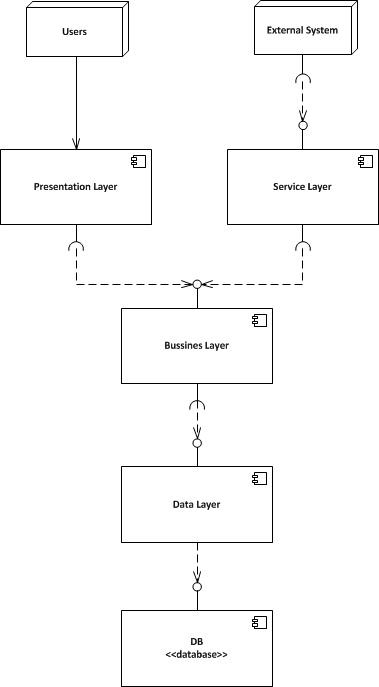
[La vista lógica permite describir el sistema en base a abstracciones fundamentales del diseño orientado a objetos para dar soporte a los requerimientos funcionales. En un enfoque top-down, se comienza por descomponer el sistema en un conjunto de subsistemas “grandes”, como ser las “capas” (*layers*) si se utiliza una arquitectura en capas, y a partir de ellos se realizan sucesivos refinamientos hasta llegar a las unidades lógicas más pequeñas.

A modo de ejemplo, se presenta parte de la vista lógica especificada por el Grupo 6 de la edición 2012 de TSI1 para la plataforma INDIGNADOFRAMEWORK.]

## Estilo Arquitectónico.

[En esta sección se describe el estilo arquitectónico elegido para el sistema (capas estricto, capas no estricto, etc.). En el lenguaje de modelado UML 2, se utiliza el Diagrama de Componentes para representar componentes lógicos, creados en tiempo de diseño, y no componentes físicos (ejecutables, bibliotectas, etc.) creados en tiempo de implementación, como sucede con el Diagrama de Componentes de UML 1. Se aconseja utilizar diagramas de componentes para representar subsistemas (considerar el clasificador <<subsystem>> como alternativa a <<component>>.]

En el diseño arquitectónico de INDIGNADOFRAMEWORK se ha decidido utilizar un estilo arquitectónico en capas. En este esquema las capas más altas utilizan servicios definidos en las capas más bajas. También se ha diseñado una capa de servicios (Service Layer) con el fin de integrarla aplicación con otras instalaciones INDIGNADOFRAMEWORK.



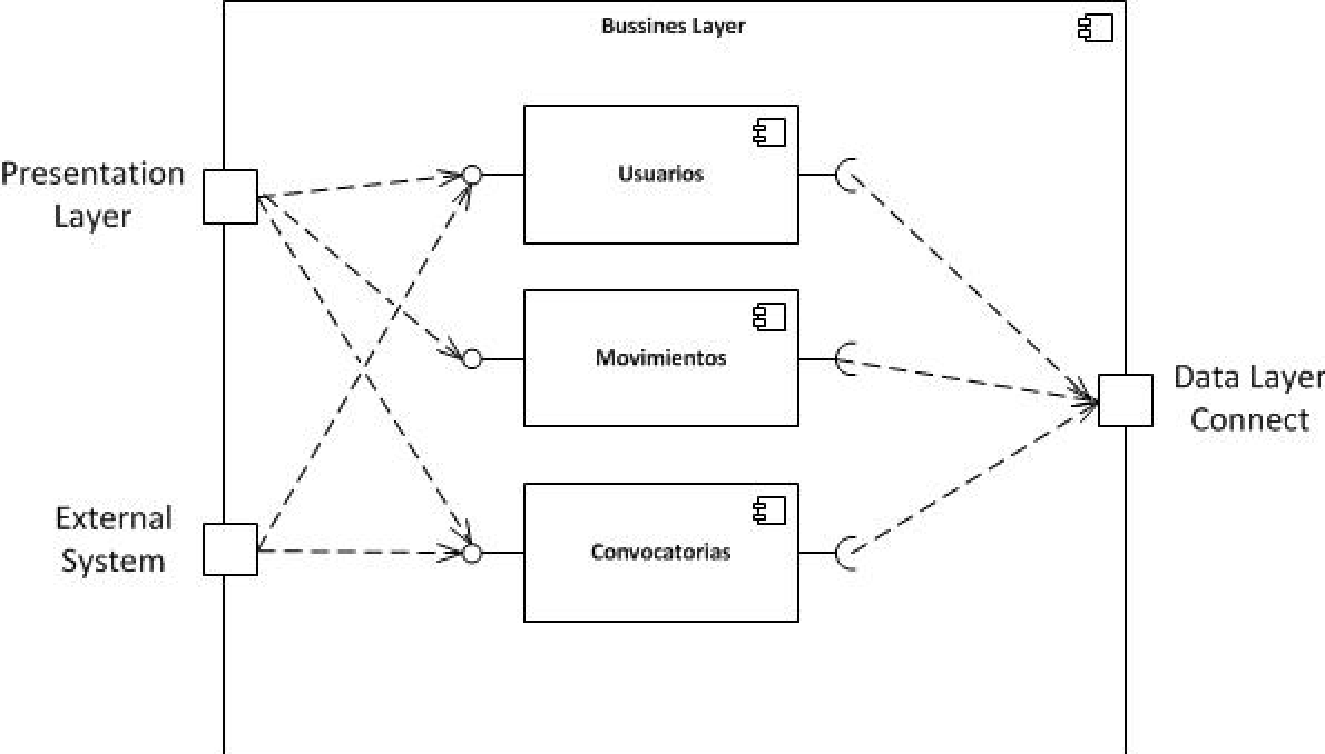
## 

## Subsistemas

[En esta sección se describe cada uno de los subsistemas. El diagrama de componentes muestra en detalle los componentes que corresponden a ese subsistema. Para cada componente, deben detallarse las interfaces que implementa.]

### **Business Layer.**

#### Diagrama de Componentes.



#### ***Descripción.***

Implementa la funcionalidad principal de la aplicación INDIGNADOFRAMEWORK. Encapsula la lógica de negocio relevante para la aplicación. Consiste en componentes, los cuales exponen (en algunos casos) interfaces para que otros utilicen. Esta componente controla el acceso a los servicios de negocio desde otras capas, la publicación de los servicios de negocio mencionados e invocación de la capa de persistencia. Se distinguen tres componentes principales en la capa de negocio:

*Usuarios*:

Aquí se desarrolla toda la lógica de negocio referente al manejo de usuarios en el sistema. Algunas de sus funcionalidades son el registro, manejo de contenidos publicados y manejo de perfil de usuario.

*Movimientos*:

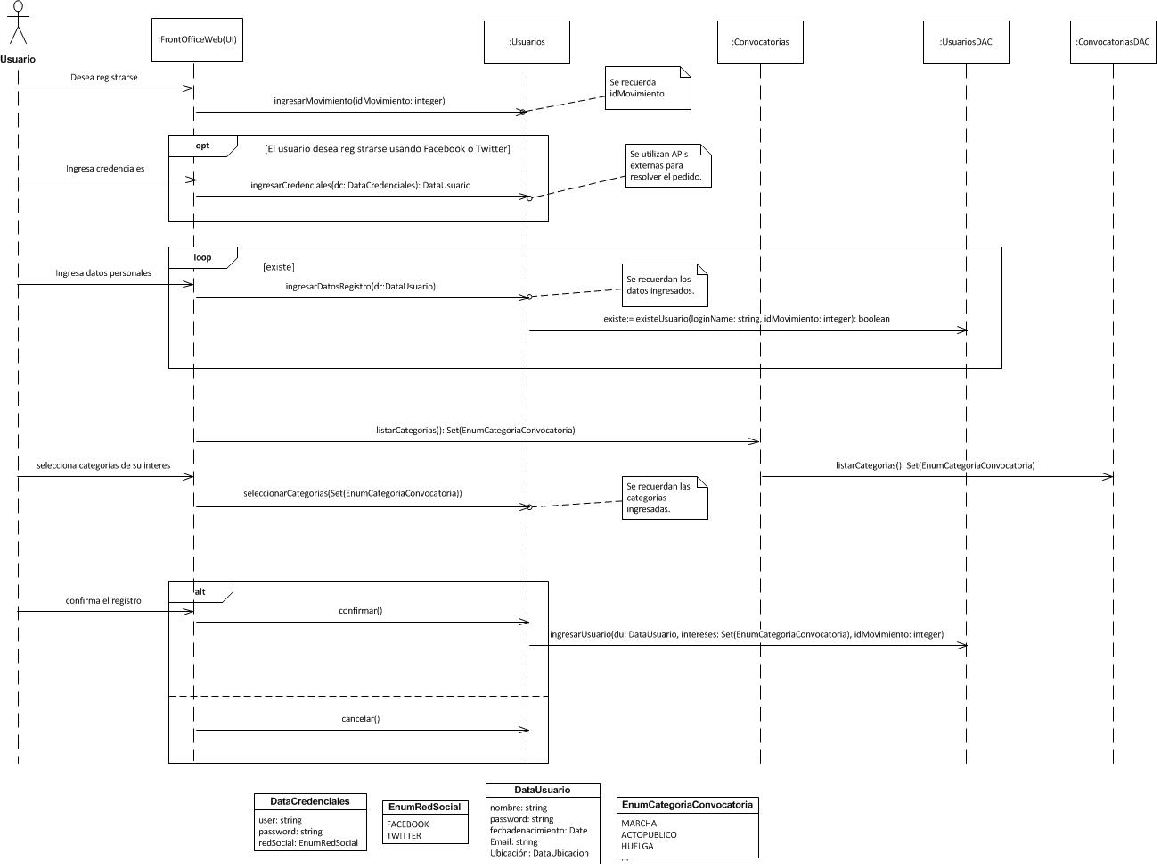
Aquí se desarrolla toda la lógica de negocio referente al manejo de los movimientos de protesta. Se en-foca en las funcionalidades del módulo BackOffice para el manejo de los administradores. Algunas de sus funcionalidades son la creación de movimientos y su administración, edición de estilos y manejo de galería de imágenes.

*Convocatorias:*

Aquí se desarrolla toda la lógica de negocio referente al manejo de convocatorias en el sistema. Algunas de sus funcionalidades son la creación y gestión de convocatorias.

## Diagramas de Interacción

[En esta sección se incluyen el o los diagramas de interacción de uno o más escenarios de cada caso de uso crítico que se considere apropiado especificar con mayor detalle. Se recomienda utilizar el Diagrama de Secuencia de UML, a nivel de componentes y no de clases.]



# Vista de Distribución (Deployment).

[En la vista de distribución o deployment se plantean uno o varios escenarios de distribución de los componentes en *tiers* o nodos. Se utiliza el Diagrama de Deployment de UML que permite mostrar la estructura del escenario en base a nodos, conexiones entre nodos y componentes de cada nodo. Un nodo puede ser un elemento de hardware (un host, un router, etc.) o un elemento de software (un servidor de aplicaciones JEE, por ejemplo). Para distinguir los nodos físicos (hardware) se utiliza el estereotipo <<device>>.

Para cada escenario planteado, interesa particularmente la descripción de los requerimientos no funcionales que justifican dicho escenario.

Opcionalmente, pueden describirse los nodos y conexiones presentes. De los nodos pueden especificarse requerimientos de software y hardware (sistema operativo, procesador, memoria, almacenamiento secundario, etc.) y de las conexiones los protocolos de comunicación, ancho de banda, etc.

A modo de ejemplo, se presenta parte de la vista de distribución especificada por el Grupo 6 de la edición 2012 de TSI1 para la plataforma INDIGNADOFRAMEWORK.]

## ***Escenario 1***

### **Descripción**

[Descripción general del escenario.]

### **Nodos**

[Descripción de cada nodo.]

### **Conexiones**

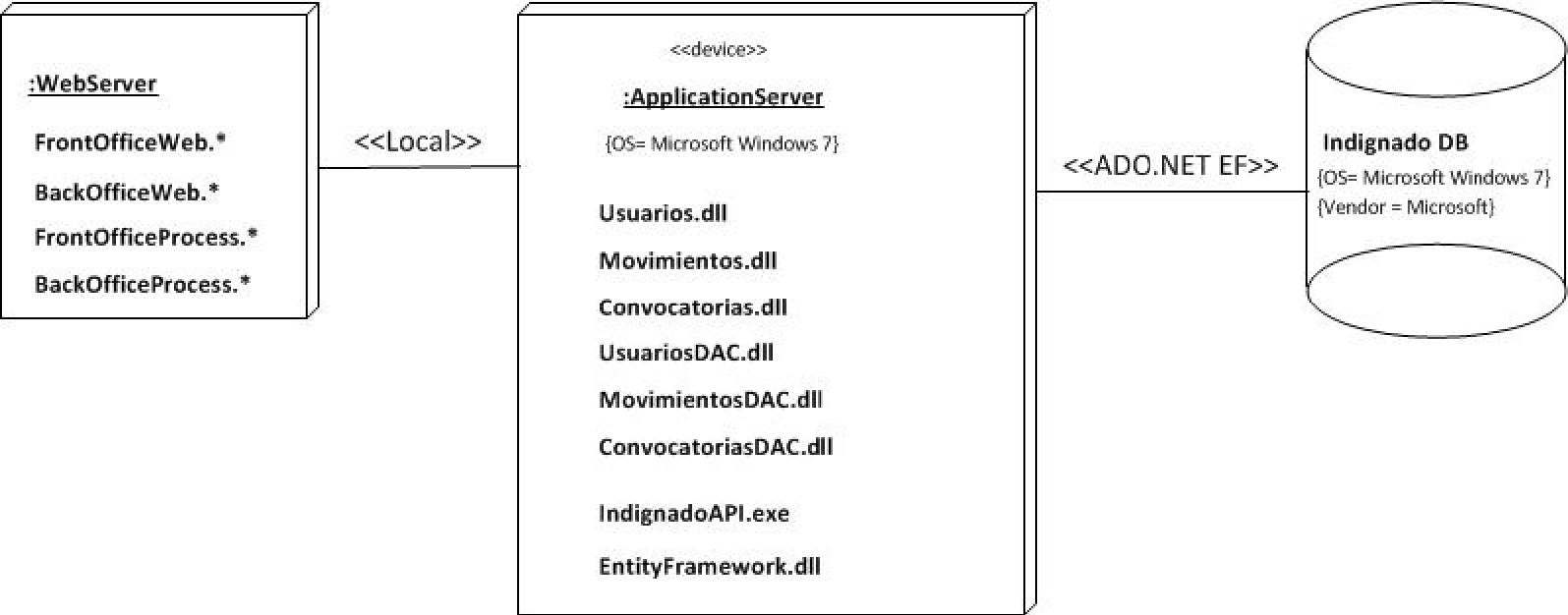
[Descripción de las conexiones.]

# Vista de Implementación.

[La vista de implementación se enfoca en los componentes en tiempo de ejecución que forman el sistema (ejecutables, archivos de clases, bibliotecas, frameworks, etc.), que son la implementación de los componentes lógicos (provenientes de Diseño).

Dentro de esta vista, interesa mostrar las dependencias entre componentes implementados (utilizando los “artefactos”de UML). También pueden mostrarse la distribución de los artefactos en los nodos. En ambos casos se utiliza el Diagrama de Deployment de UML.

A modo de ejemplo, se presenta parte de la vista de implementación especificada por el Grupo 6 de la edición 2012 de TSI1 para la plataforma INDIGNADOFRAMEWORK.]



# Referencias

1. Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives. N. Rozanski, E. Woods. Addison-Wesley, 2005
2. Software Architecture in practice, Second Edition. L. Bass, P. Clemens, R. Kazman. Addison-Wesley, 2003
3. An Introduction to Software Architecture. D. Garlan, M. Shaw. 1994. <http://www.cs.cmu.edu/afs/cs/project/able/ftp/intro_softarch/intro_softarch.pdf>
4. Architectural Blueprints — The “4+1” View Model of Software Architecture. Kruchten, Philippe. 1995. <http://www.cs.ubc.ca/~gregor/teaching/papers/4+1view-architecture.pdf>
5. Software Architecture Links. Bredemeyer Consulting. <http://www.bredemeyer.com/links.htm>
6. Grupo 6, TSI1 edición 2012. Documento de Arquitectura. 2012.