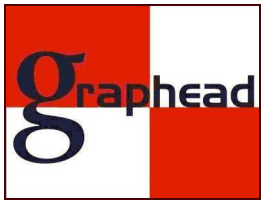
Facultad de Ingeniería

Pasantía

Evaluación de Arquitecturas de Software

Aplicando ATAM

Evaluación de Graphead



Responsable	
Andrea Delgado	
Integrantes	
Alberto Castro	3.239.979-8
Martín Germán	3.217.050-4

Versión 5.0
Documento 2/3

ÍNDICE

1 INTRODUCCIÓN

Una arquitectura de software de un programa o sistema de computación es la estructura o las estructuras del sistema, que contienen componentes de software, las propiedades externamente visibles de dichos componentes y las relaciones entre ellos.

¿Cómo se puede estar seguro que la arquitectura elegida es la correcta para un software? No es una pregunta fácil, lo que si se sabe es que la arquitectura es una pieza fundamental de cualquier software. La arquitectura es quien define los atributos de calidad de un sistema. Entonces, si las decisiones arquitectónicas determinan los atributos de calidad del sistema, es posible evaluar las decisiones arquitectónicas con respecto a su impacto sobre dichos atributos.

Una mala arquitectura puede llevar a un proyecto al fracaso. Todos los requerimientos de calidad pueden quedar insatisfechos. Una evaluación es útil para entender si el sistema cumple con los requerimientos de calidad y comportamiento.

En términos concretos, la evaluación de la arquitectura produce un informe, la forma y contenido del mismo varía según el método utilizado. En particular, produce repuestas a dos tipos de preguntas:

- ¿Es esta arquitectura adecuada para el sistema para la cual fue diseñada?
- ¿Cuál de dos o más arquitecturas propuestas es la más adecuada para el sistema?

Un resultado que también produce la evaluación de una arquitectura es la captura y priorización de las metas que la arquitectura debe cumplir para poder ser considerada adecuada.

Este informe pretende responder a las interrogantes planteadas anteriormente, cuando dichas preguntas se le realizan al sistema *Graphead*. La forma en que se responderá a las cuestiones de la arquitectura de dicho producto, será mediante el método de evaluación elegido, el Architecture Tradeoff Analysis Method (ATAM), quien guiará la evaluación, y cuyas salidas contiene el presente documento.

El objetivo principal de este trabajo es verificar que el producto cumpla con los requerimientos del cliente, el logro de las metas del negocio y la calidad total del software.

1.1 Características del Producto

El software *Graphead* será utilizado para la administración de procesos y proyectos de una organización. El objetivo del mismo es generar el ambiente necesario para el gerenciamiento de los procesos y los proyectos de una organización, y además permitir hacerlo en forma simple mediante una herramienta gráfica, que será parte del sistema. También cuenta con una simple visualización de los procesos y proyectos por medio de páginas html y documentos imprimibles.

El sistema contempla que los distintos procesos sean adaptados a la medida de cada organización, pero que además sean adaptados dentro de la misma a lo largo del tiempo, lo que lleva a distintas versiones (cambios documentados) del mismo.

El diseño de este sistema esta enfocado a ofrecer una interfaz de usuario amigable que facilite el manejo de los procesos y los elementos que lo componen.
La herramienta cuenta con una aplicación de escritorio que permite diseñar gráficamente un modelo de proceso, esto se traduce en:

- Gestión de Actividades
- Gestión de Agenda

- Gestión de Definiciones
- Gestión de Disciplinas
- Gestión de Entregables
- Gestión de Fases
- Gestión de Guías
- Gestión de Herramientas
- Gestión de Iteraciones
- Gestión de Roles
- Gestión de Vistas

A su vez, brinda soporte de versionado tanto para elementos dentro de un modelo de proceso, como también para procesos propiamente dichos. También brinda la funcionalidad de exportar e importar procesos, permitiendo así la migración de estos entre diferentes PC. Otra característica de la aplicación, es la posibilidad de generar un sitio Web representativo de un modelo de proceso, como también un documento en formato imprimible del mismo (pdf). Además, es multiplataforma, multilenguaje y de distribución gratuita.

1.2 Resumen del ATAM

La parte principal del ATAM consiste de nueve pasos. Estos pasos se dividen cuatro grupos:

- Presentación, donde la información es intercambiada.
- Investigación y análisis, donde se valoran los atributos claves de calidad requeridos, uno a uno con las propuestas arquitectónicas.
- Pruebas, donde se revisan los resultados obtenidos contra las necesidades relevantes de los stakeholders.
- Informes, donde se presentan los resultados del ATAM.

1.2.1 Presentación

1. ***Presentar el ATAM***

   El líder del equipo de evaluación describe el método a los participantes, fija las expectativas y responde las preguntas que puedan surgir.

2. ***Presentar las pautas del negocio***

   Un representante del proyecto (por ejemplo, el director de proyecto o el cliente del sistema) describe las metas del negocio que motivan el esfuerzo de desarrollo y que se convertirán en las pautas primordiales de la arquitectura (por ejemplo, alta disponibilidad o alta seguridad).

3. ***Presentar la arquitectura***

   El arquitecto describe la arquitectura, enfocándose en como esta sigue las pautas del negocio.

1.2.2 Investigación y análisis

4. ***Identificar las propuestas arquitectónicas***

   Las propuestas arquitectónicas son identificadas por el arquitecto, pero no son analizadas.

5. ***Generar el árbol de utilidad de los atributos de calidad***

   Los atributos de calidad que comprometen la utilidad del sistema (performance, availability, entre otros) son obtenidos, especificados en escenarios (con estímulos y respuestas) y priorizados.

6. ***Analizar las propuestas arquitectónicas***

   Basados en los escenarios de mayor prioridad identificados en el paso 5, las propuestas arquitectónicas que cumplen con estos escenarios, son obtenidas y analizadas (por ejemplo, una propuesta arquitectónica que logra una meta de performance, será objeto de un análisis de performance).
   Durante este paso los riesgos arquitectónicos, los no riesgos, los sensitivity points y tradeoff points son identificados.

1.2.3 Pruebas

7. ***Lluvia de ideas y priorización de escenarios***

   Un gran conjunto de escenarios es obtenido por todos los stakeholders. Este conjunto es priorizado mediante votación.

8. ***Analizar las propuestas arquitectónicas***

   En este paso se reitera las actividades del paso 6 utilizando el ranking de escenarios del paso 7. Estos escenarios se consideran casos de prueba para confirmar el análisis realizado hasta ahora. Este análisis puede descubrir nuevas propuestas arquitectónicas, riesgos, no riesgos, sensitivity points y tradeoff points, que son documentados.

1.2.4 Informes

9. ***Presentar los resultados***

   Basados en la información recogida durante el ATAM (propuestas, escenarios, árbol de utilidad, riesgos, no riesgos, sensitivity points y tradeoff points), el equipo de ATAM presenta los resultados a los stakeholders.

2 PAUTAS DEL NEGOCIO

2.1 Requerimientos priorizados

2.1.1 Procesos

Un proceso para el desarrollo de software es un conjunto de procedimientos, técnicas, herramientas, y un soporte documental que ayuda a los desarrolladores a producir nuevo software.

2.1.1.1 Alta de Procesos

Se debe manejar información de los procesos, mantener su nombre y descripción. Se pueden crear nuevos procesos.

2.1.1.2 Copias de elementos de un proceso a otro

Es de interés poder copiar partes de un proceso a otro con todo lo que la parte esta relacionada, esto implica que parte de un proceso, por ejemplo una actividad, puede ser copiada de un proceso a otro copiándose no solo el nombre y la descripción de la actividad, sino que además se hará una copia de todos los elementos con los que esta relacionada "hacia abajo" como son entregables, roles, entre otros.

2.1.2 Actividades

Una actividad es algo que un rol hace y eso proporciona un resultado significativo en el contexto del proyecto. Las actividades están estrechamente relacionadas a los entregables. Los entregables son la entrada y salida de las actividades, y el mecanismo mediante el cual se comunica la información entre las actividades.

2.1.2.1 Alta de Actividades

El sistema contara con una herramienta para crear actividades, en ella se ingresaran los datos necesarios y estas actividades creadas podrán ser usadas para el diseño de los procesos.

2.1.2.2 Agregar actividades a procesos

Los usuarios podrán agregar actividades a los distintos procesos.

2.1.3 Roles

Un rol define el comportamiento y responsabilidades de un individuo, o un conjunto de individuos que trabajan juntos como un equipo, dentro del contexto de una organización de ingeniería de software. Un integrante del proyecto puede cumplir varios roles así como también, por definición, un rol puede ser asumido por más de un integrante. Los roles tienen un conjunto de actividades cohesivas que realizan. Estas actividades están estrechamente relacionadas y funcionalmente acopladas, y se realizan mejor por el mismo individuo. Un rol por ejemplo es el de arquitecto, que es quien se ocupa de establecer la estructura para cada punto de vista de la arquitectura.

2.1.3.1 Alta de Roles

El sistema permitirá crear roles dentro del diseño del proceso.

2.1.4 Entregables

Un entregable es todo tipo de información creada, producida, cambiada o usada por el proceso. Un entregable es un producto del proceso: las personas de los distintos roles usan entregables para realizar actividades, y producen entregables en la realización de las actividades.

2.1.4.1 Alta de Entregables

El sistema permitirá la creación, modificación y borrado de entregables.

2.1.5 Plantillas

Una plantilla es un documento que está asociado a un entregable, que muestra el formato que el mismo debe seguir.

2.1.5.1 Crear asociaciones entregable – plantilla

El sistema permitirá asociar a un entregable una plantilla.

2.1.6 Generación del sitio Web del proyecto

En el momento que el proyecto es creado por un usuario administrador este podrá generar un sitio Web en forma automática mediante el cual se realizará la gestión del mismo por el grupo al que el proyecto pertenezca. El sitio será de acceso restringido a los integrantes del proyecto y mediante el podrán subir los entregables según correspondan por semana (quedando determinada de esta manera la agenda real del proyecto). Mediante este sitio además podrán realizar aclaraciones y comentarios, e ingresar métricas.

2.1.7 Generación del proceso en formato imprimible

Se deberá poder generar para un proceso su versión imprimible en formato estándar (pdf).
La generación deberá ser automática.

2.1.8 Manejo de Versiones

El sistema guardará las distintas versiones, de manera que el usuario puede volver en cualquier momento a una versión anterior de un proceso. La versión vieja se guarda y se podrá consultar y reutilizar.

2.1.8.1 Manejo de Versiones para procesos

El sistema permitirá al usuario definir versiones de procesos. Cuando un usuario desea realizar un cambio pero pretende mantener el proceso anterior, por ejemplo cuando se produce un cambio de año en el caso de proyectos de ingeniería de software, el usuario indica al sistema que desea versionar el proceso indicando la razón del cambio.

2.1.9 Creación de agenda

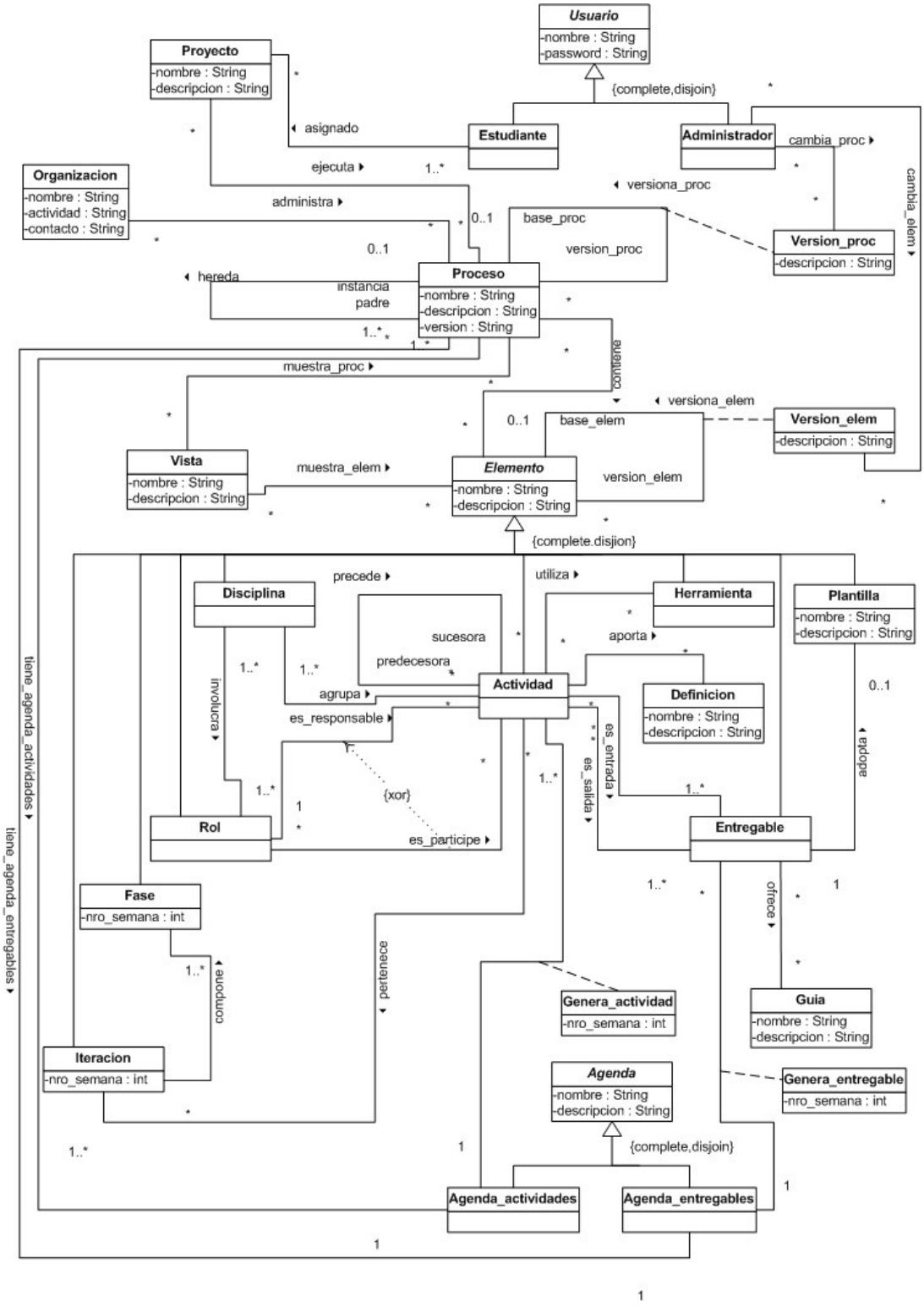
El usuario administrador podrá generar una agenda actividades, esto lo hará asociando a las actividades incluidas en el proceso una semana para la cual será planeada. A su vez esta generará automáticamente una agenda de entregables según los entregables asociados a cada actividad. Esto implica asignar a cada semana en la que se dividió el proyecto un conjunto de entregables de forma que un entregable este en una semana si es salida de una actividad que se definió para esa semana.

2.1.10 Se deben poder importar y exportar procesos

Se debe poder exportar e importar procesos tanto en construcción como construidos, permitiéndole al usuario hacer modificaciones a este desde distintas copias del sistema.

2.2 Dominio de aplicación

2.2.1 Modelo de Dominio



2.2.2 Restricciones al Modelo de Dominio

- Si un proceso es ejecutado por un proyecto, este no puede tener instancias.
- Ningún proceso que es instancia puede ser instanciado por otro proceso (solo puede será ejecutado por un proyecto).
- Si un proceso tiene instancia, los elementos de esta instancia no deben estar incluidos en los elementos del mismo.

- Ningún proceso puede ser instancia de sí mismo.
- Ningún proceso puede ser base de sí mismo.
- Ningún proceso puede ser versión de sí mismo.
- Si un proceso B es versión de un proceso A y un proceso C es versión de B, entonces C no puede ser versión de A (no hay ciclos en la relación "Version_proc").
- Ningún elemento puede ser base de sí mismo.
- Ningún elemento puede ser versión de sí mismo.
- Si un elemento B es versión de un elemento A y un elemento C es versión de B, entonces C no puede ser versión de A (no hay ciclos en la relación "Version_elem").
- Las versiones de elementos deben ser del mismo tipo del elemento que versionan. Por ejemplo: una fase debe ser versión de una fase y solamente de una fase.
- Ninguna actividad puede ser predecesora de sí misma.
- Ninguna actividad puede ser sucesora de sí misma.
- Si una actividad B es sucesora de una actividad A y una actividad C es sucesora de B, entonces C no puede ser predecesora de A (no hay ciclos en la relación "precede").
- Una disciplina debe involucrar roles que son responsables de actividades o (exclusivo) debe involucrar roles que son participantes de actividades, que la misma agrupa.
- Un proceso tiene una agenda de actividades generada sobre la base de actividades contenidas en el mismo.
- Un proceso tiene una agenda de entregables generada sobre la base de entregables contenidos en el mismo.

2.3 Casos de Uso priorizados

2.3.1 Alta de Procesos

2.3.1.1 Descripción

Un administrador crea un nuevo proceso ingresando su nombre y si lo desea su descripción.

2.3.1.2 Pre-condiciones

No tiene.

2.3.1.3 Post-condiciones

Queda registrado el proceso en el sistema con el nombre especificado.

2.3.1.4 Requerimientos especiales

No tiene.

2.3.2 Cargar Procesos:

2.3.2.1 Descripción:

El usuario recupera un proceso que estaba diseñando, buscando entre los procesos en diseño existentes.

2.3.2.2 Pre-condiciones

El administrador debe haber pasado la instancia de autenticación.

2.3.2.3 Post-condiciones

No tiene.

2.3.2.4 Requerimientos especiales

No tiene.

2.3.3 Alta de Actividades

2.3.3.1 Descripción

Un administrador selecciona una disciplina y crea una actividad dentro de la misma. Esta actividad queda asociada al proceso que está diseñando, además de cargar los datos, mediante el uso de la herramienta gráfica se puede:

- Asociar Roles a la actividad.
- Asociar Definiciones a la actividad.
- Asociar Guías a la actividad.
- Asociar Entregables a la actividad (esta relación puede modelar que el Entregable es salida de la Actividad o entrada de ella).
- Asociar con una Herramienta del proceso.

2.3.3.2 Pre-condiciones

El administrador debe haber pasado la instancia de autenticación.

2.3.3.3 Post-condiciones

Queda registrada la actividad en el sistema con el nombre especificado, las relaciones establecidas y con número de versión 0.

2.3.3.4 Requerimientos especiales

No tiene.

2.3.4 Agregar elemento existente al proceso

2.3.4.1 Descripción

El usuario realiza clic derecho sobre el tipo de componente que desea agregar y elige la opción "Agregar Componente", se despliegan todos los componentes disponibles de ese tipo y el usuario selecciona uno agregándolo así a su proceso.

2.3.4.2 Pre-condiciones

El administrador debe haber pasado la instancia de autenticación.

2.3.4.3 Post-condiciones

Queda registrado en el sistema que el elemento pertenece al proceso especificado con el contenido elegido.

2.3.4.4 Requerimientos especiales

No tiene.

2.3.5 Dar de alta un rol

2.3.5.1 Descripción

Un administrador registra un nuevo rol en el sistema, puede también asociar el rol que crea a una actividad o un entregable. Como resultado el rol queda registrado en el sistema.

2.3.5.2 Pre-condiciones

El administrador debe haber pasado la instancia de autenticación.

2.3.5.3 Post-condiciones

Queda registrado el rol en el sistema con el nombre especificado, las relaciones establecidas.

2.3.5.4 Requerimientos especiales

No tiene.

2.3.6 Alta de Entregable

2.3.6.1 Descripción

El usuario crea un Entregable y queda agregado al proceso que está diseñando, además puede agregar relaciones con Roles del proceso, pero esto se realiza en el contexto de Diseño de Actividades.

2.3.6.2 Pre-condiciones

El administrador debe haber pasado la instancia de autenticación.

2.3.6.3 Post-condiciones

Queda registrado el entregable en el sistema con el nombre especificado, la descripción y una plantilla asociada.

2.3.6.4 Requerimientos especiales

No tiene.

2.3.7 Asociar una Plantilla a un Entregable

2.3.7.1 Descripción

El usuario selecciona un entregable y luego elige una plantilla para asociarle al mismo.

2.3.7.2 Pre-condiciones

El administrador debe haber pasado la instancia de autenticación.

2.3.7.3 Post-condiciones

Al entregable seleccionado por el administrador se le asocia la plantilla elegida.

2.3.7.4 Requerimientos especiales

No tiene.

2.3.8 Generar Sitio Web

2.3.8.1 Descripción

Dado un modelo de proceso, genera contenido html representativo del mismo. Además al sitio Web se podrá agregar una sección para que los estudiantes puedan subir entregables.

2.3.8.2 Pre-condiciones

El administrador debe haber pasado la instancia de autenticación

2.3.8.3 Post-condiciones

Se tiene el sitio Web generado en el sistema.

2.3.8.4 Requerimientos especiales

No tiene.

2.3.9 Generar formato imprimible

2.3.9.1 Descripción

Genera la versión imprimible de un modelo de proceso. El documento generado será pdf.

2.3.9.2 Pre-condiciones

El administrador debe haber pasado la instancia de autenticación.

2.3.9.3 Post-condiciones

Se tiene el documento pdf generado en el sistema.

2.3.9.4 Requerimientos especiales

No tiene.

2.3.10 Crear una nueva versión de un proceso

2.3.10.1 Descripción

El usuario realiza el cambio en el proceso e ingresa una descripción y motivo del cambio. Queda guardada una nueva versión del proceso que puede ser recuperada en cualquier momento y verse como estaba antes de cambiar.

2.3.10.2 Pre-condiciones

El administrador debe haber pasado la instancia de autenticación.

2.3.10.3 Post-condiciones

Queda registrada en el sistema una nueva versión par el proceso seleccionado, con la descripción del cambio.

2.3.10.4 Requerimientos especiales

No tiene.

2.3.11 Crear una agenda para un proceso

2.3.11.1 Descripción

El usuario asocia actividades del proceso a las semanas que dura el proceso, esto genera una agenda de actividades.

2.3.11.2 Pre-condiciones

El administrador debe haber pasado la instancia de autenticación.

2.3.11.3 Post-condiciones

Queda registrada la nueva agenda para el proceso y la información referente a la misma.

2.3.11.4 Requerimientos especiales

No tiene.

2.3.12 Importación y Exportación de procesos

2.3.12.1 Descripción

Un usuario administrador podrá exportar procesos, se entiende por esto que podrá guardar el proceso en forma externa al sistema de forma que este se haga portable a otras instancias de la aplicación donde podrá ser modificado.
La aplicación permitirá importar procesos con sus componentes y utilizarlos como si hubiesen sido creados en esa instancia de la aplicación.

2.4 Stakeholders

2.4.1 Cliente

2.4.1.1 Grupo de Ingeniería de Software

Se interesa en los procesos de producción de software y su relación con la obtención de productos de calidad con un costo controlado y en los plazos requeridos. Es en este contexto que considera relevantes los modelos de proceso, la obtención y gestión de requerimientos, la evaluación de arquitecturas, la verificación y las métricas del software. El estudio de estos temas se realiza en proyectos conjuntos con la industria y en un laboratorio en el que estudiantes alcanzan una primera experiencia de envergadura en la producción de software.

Áreas de Interés:

- Procesos de desarrollo de software
- Requerimientos
- Diseño
- Verificación
- Gestión de Proyectos de Software
- Calidad en el Software
- Mejora de Procesos

2.4.2 Usuario final

2.4.2.1 Administrador

Accede a todas las funcionalidades del sistema pero dependiendo del perfil al que este asociado tiene distintos accesos a los datos del sistema.

Los perfiles definen:

- Poder diseñar solo procesos propios y no tener acceso a los otros.
- Ver todos los procesos y poder crear nuevos, modificar o borrar los existentes.
- Crear usuarios y modificar el perfil asociado a un usuario.

2.4.2.2 Estudiante

Esta asignado a un proyecto y tiene acceso a la información del mismo y esta autorizado a realizar los entregables que le correspondan.

2.5 Restricciones

2.5.1 Restricciones del Negocio

2.5.1.1 Licenciamiento

- Todo software utilizado en la producción del sistema debe de ser de licencia libre.

2.5.2 Restricciones Técnicas

- Las páginas suministradas en forma automática por el software deben ser visibles y utilizables en los siguientes navegadores: Internet Explorer.
- El software debe ser utilizable en los siguientes sistemas operativos: Unix y Windows.
- Se debe trabajar sobre la base de datos MySql.
- Se debe utilizar la VM de java en su versión 1.4.1.

2.6 Atributos de Calidad priorizados

2.6.1 Performance

2.6.1.1 Respuesta de página Web de proyectos

El tiempo de respuesta de la página Web de los proyectos deberá estar en parámetros aceptables para una aplicación Web.

2.6.2 Usabilidad

2.6.2.1 Simplicidad de uso

Se requiere que las interfaces gráficas definidas permitan un manejo simple de la gestión de proyectos.

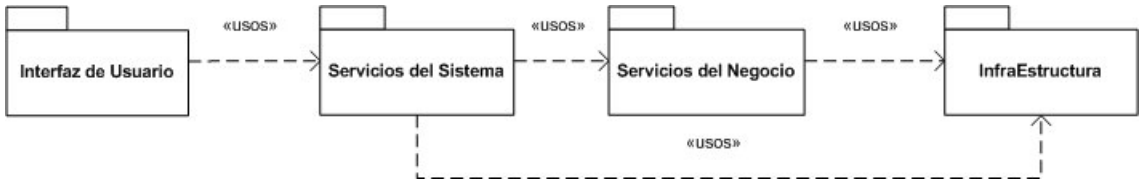
2.6.2.2 Simplicidad en la presentación de información

Sobre los documentos en formatos de imprimibles y los en formato html es un requerimiento que estos sean de fácil recorrida y visualización por los usuarios, en particular fácil navegación en el caso de las paginas html.

3 PROPUESTAS ARQUITECTÓNICAS

3.1 Descomposición en Capas

La arquitectura está organizada con un estilo de capas relajado. A continuación se muestra un diagrama de esta arquitectura:



El estilo de capas relajado implica que una capa puede utilizar servicios de otra, más de un nivel por debajo de ella. En este caso: la capa de Servicios de Sistema utiliza servicios de Infraestructura.
Cada capa determina un rol para los módulos que residen en ella.
A continuación se especifica el propósito particular y el contenido de cada capa componente de la arquitectura del sistema desarrollado.

3.1.1 Interfaz de Usuario

La Interfaz de Usuario tiene como objetivo el manejo de la lógica del usuario. Está conformada por el conjunto de componentes de *JGraph* y por módulos que encapsulan la lógica de los casos de uso.

3.1.2 Servicios de Sistema

Los Servicios del Sistema representan los servicios básicos que debe proveer el sistema; estos servicios son directamente utilizados por los módulos de la capa superior.
Esta capa representa la entrada a la lógica del sistema. Se cuenta con uno o más módulos por cada uno de los subsistemas identificados. Estos módulos ofrecen las interfaces requeridas por cada caso de uso asociado al subsistema y los controladores que las implementan.

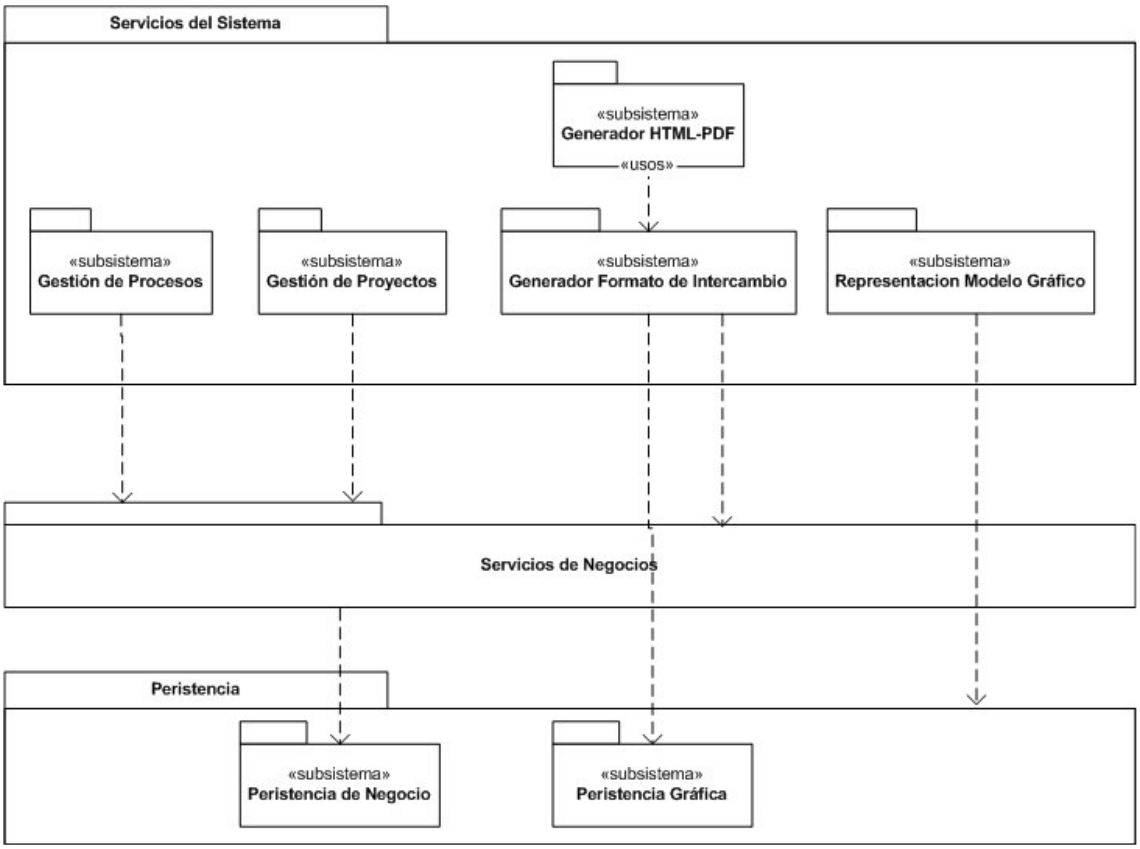
3.1.3 Servicios de Negocios

Los Servicios de Negocio son servicios de manejo de información del negocio; son servicios aun más básicos que el de la capa superior. Esto permite reutilizar estos módulos en otros subsistemas.
En esta capa, se encuentran los manejadores de los conceptos principales del modelo de dominio del problema. Estos manejadores encapsulan el acceso a los datos, ocultando para los servicios de nivel superior la forma en que los mismos son almacenados.
A diferencia de los módulos en las capas superiores los aquí presentes podrán ser compartidos por varios subsistemas.

3.1.4 Infraestructura

Capa en la cual se ubican módulos adaptadores y de servicio general, útiles para cualquiera de los subsistemas.
Aquí se encuentra el encapsulador del DBMS utilizado.

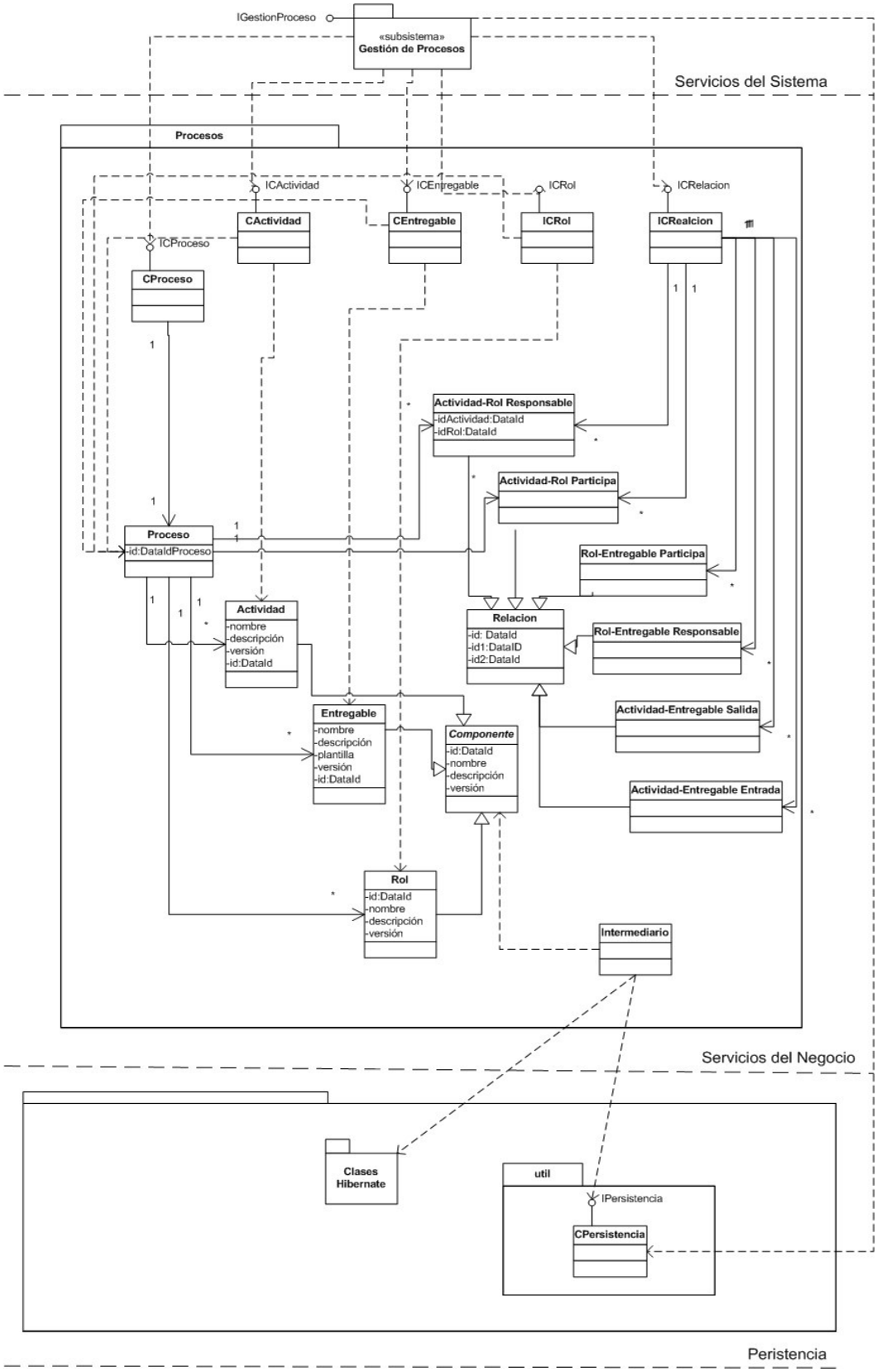
3.2 Descomposición en Subsistemas



3.2.1 Capa de Servicios del Sistema

3.2.1.1 Gestión de Procesos

Resuelve todas las funcionalidades referentes a la creación, modificación y eliminación de modelos de procesos, como también de sus componentes.



3.2.1.2 Gestión de Proyectos

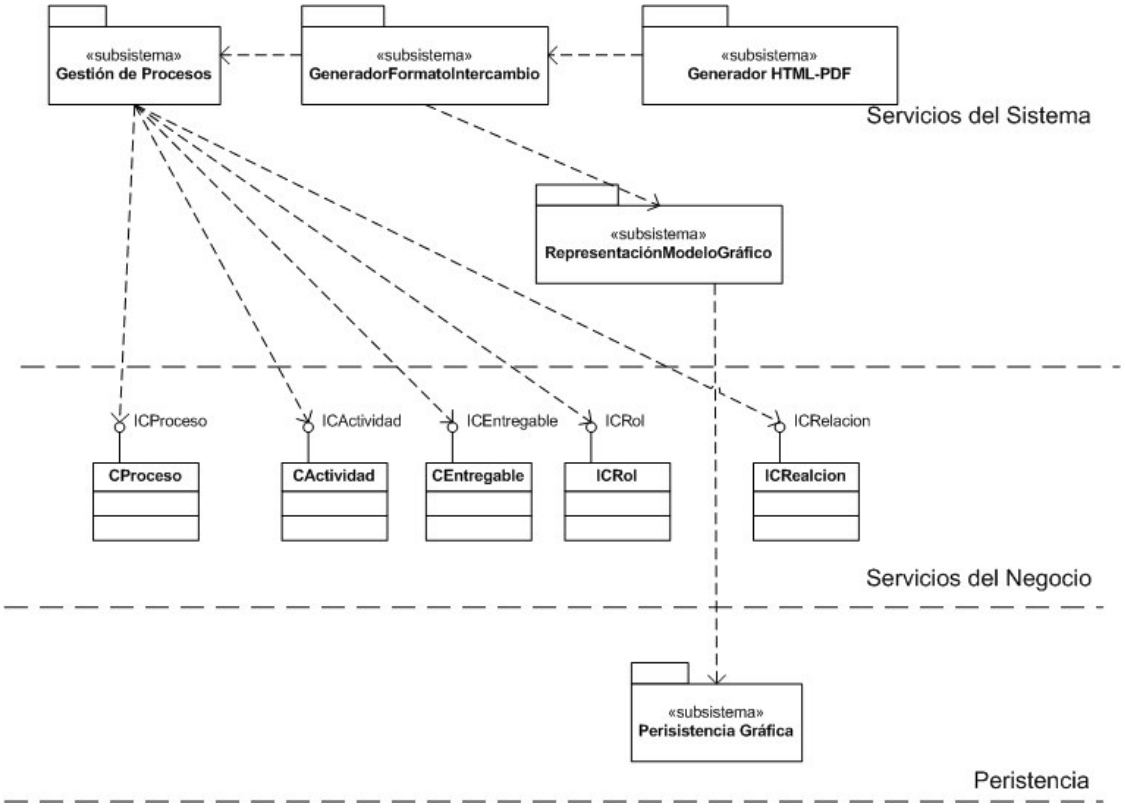
Se encarga del manejo a todo lo concerniente a un proyecto, tales como grupos, integrantes, director, entre otros.

3.2.1.3 Generador de Formato de Intercambio

Es el encargado de generar información acerca de un modelo de proceso en formato xml, el cual será interpretado por el subsistema "Generador html-pdf".

3.2.1.4 Generador html-pdf

A partir de un formato xml representativo de un modelo de proceso, genera el conjunto de paginas html que conforman el sitio Web para el mismo, como también un documento pdf.



3.2.1.5 Representación Modelo Gráfico

Provee las funcionalidades necesarias para atender las distintas opciones de guardado de datos que representan la vista gráfica del editor de modelos de procesos.

3.2.2 Capa de Persistencia

3.2.2.1 Persistencia Negocio

Provee las funcionalidades necesarias para guardado de datos representativo del las entidades del negocio del sistema.

3.2.2.2 Persistencia Gráfica

Provee las funcionalidades necesarias para guardado de datos relacionados con la vistas gráfica de la aplicación.

3.3 Cliente – Servidor

Se utiliza el patrón cliente – servidor para resolver el acceso de los usuarios a las páginas Web del Modelo de Proceso, generadas por el Sistema. Existe un servidor Web que contiene dichas páginas, y los usuarios acceden a ellas mediante un browser (cliente Web).

3.4 Consideraciones

3.4.1 Cliente – Servidor

El patrón es resuelto por software de terceros.

3.4.2 Persistencia

- Para el manejo de los datos se utiliza el DBMS *MySql*.
- El mapeo entre los conceptos de la lógica del Sistema (donde se utiliza el paradigma orientado a objetos) y los de la base de datos (donde se utiliza el paradigma relacional) es realizado por el framework *Hibernate* (mapea clases java a tablas de base de datos).

3.4.3 Generador xml

Se utilizó *Castor* para realizar para realizar el mapeo de las clases java a xml, y a partir de un archivo xml dado (con la especificación de las clases java), generar las clases java correspondientes.

3.4.4 Generador web

Para generar el sitio web del proceso, con html estático, se utilizó *Xalan*. Este a partir de un archivo xml (el creado por Castor) genera el html correspondiente, aplicando a dicho xml una transformación xsl.

3.4.5 Generador pdf

El formato de archivo imprimible elegido para representar el proceso fue pdf. Este se genera a partir del mismo xml que se utiliza para generar el sitio web, pero aplicando una transformación xsl distinta. La generación del archivo pdf se realiza utilizando *Fop*.

3.4.6 Generador de la agenda

Para la generación de la agenda del proceso en los formatos html, pdf y xls, se utilizó la herramienta *Jasper*.

3.4.7 Herramienta visual

La interfaz gráfica de Graphead proporciona una herramienta visual para construir los procesos de manera interactiva. Para relacionar los conceptos de un proceso (como ser una actividad, un rol, entre otros) mediante un grafo, a nivel gráfico se utilizo la api de *JGraph*.

4 OBSERVACIONES DE LA DOCUMENTACIÓN

En este capítulo se comentan aspectos de la documentación relevada que se consideran importantes, antes de continuar con los siguientes pasos propuestos por ATAM.

La documentación de la arquitectura debería cubrir, según ATAM, las restricciones técnicas, otros sistemas con los cuales se debe interactuar y las propuestas arquitectónicas utilizadas para alcanzar los requerimientos de los atributos de calidad.

El ATAM centraliza el análisis de una arquitectura en el entendimiento de sus propuestas arquitectónicas. Se concentra en identificar propuestas y estilos arquitectónicos porque estos representan la manera en que la arquitectura cumple con los atributos de calidad, esto significa que se asegura que los requerimientos críticos serán alcanzados en forma predecible. Estas propuestas arquitectónicas definen las estructuras importantes del sistema, y describen como el sistema puede crecer, responder a cambios, entre otros.

Luego de analizar la documentación existente del sistema, siguiendo las premisas propuestas por ATAM, se descubrieron distintas falencias en dicha documentación que influyen negativamente en el posterior desarrollo de la evaluación.

A continuación las enumeramos:

- No existe trazabilidad entre documentos.
  - No se sabe a que requerimiento de la Especificación de Requerimientos corresponde cada caso de uso del Modelo de Casos de Uso.
  - Casos de uso que describen una misma funcionalidad se llaman de una manera en el Modelo de Casos de Uso y de otra diferente en el Alcance, en el Modelo de Diseño o en el SAD.
  - No hay correspondencia entre los componentes del SAD y los del Modelo de Diseño.
- No se documentan los componentes que encapsulan las comunicaciones con software externo.
  - En particular el uso del framework Hibernate es oscuro. No se sabe como se maneja la actualización de datos a través del mismo, cuando según el Modelo de Diseño todas las colaboraciones entre componentes allí existentes, realizan persistencia.
  - No se específica que componentes encapsulan el manejo del resto del software externo utilizado, según la presentación del producto, como ser Castor, Xalan, entre otros. En si se sabe que se utilizó software externo, pero se desconoce como se interactuó con el mismo.
- El Modelo de Diseño, así como el SAD, no refina los componentes críticos del sistema con la especificidad necesaria como para entender como se satisfacen los requerimientos. En particular, no se sabe como se alcanzan los requerimientos de los atributos de calidad.
- Los requerimientos de calidad deseados por el cliente, como ser la performance, no se encuentran especificados.
- La Documentación Técnica, es solo un listado de paquetes e interfaces. No se conoce el comportamiento de cada clase o interfaz.
- En los casos de uso relevantes para la arquitectura, considerando que estos son los pertenecientes al SAD, no se sabe porque fueron seleccionados, no se sabe que funcionalidad aportan al sistema, en si no existe una justificación para la elección de los mismos.
- En el Modelo de Diseño aparecen clases y dependencias que se desconoce de donde provienen o cual es su finalidad, directamente algunas de estas clases no son utilizadas en ningún diagrama de secuencia realizado en la documentación.

A pesar de lo anteriormente mencionado, el cliente realiza las siguientes consideraciones:

- La Especificación de Requerimientos es completa y expresa los intereses que el cliente tiene del sistema.
- El Modelo de Casos de Uso contiene todos los casos de uso del sistema a desarrollar.
- Cabe aclarar, empero de que no existe trazabilidad entre los documentos anteriores, los mismos fueron validados por el cliente.

Un aspecto que estimamos positivo de la documentación existente, es la representación definida en xml de un proceso. Encontramos, a pesar de que la estructura diseñada no es muy prolija, que la misma contiene todos los conceptos y relaciones que denotan a un proceso.

5 ENTREVISTA CON EL ARQUITECTO

Para ATAM es muy importante la participación del arquitecto en las diferentes etapas de la evaluación, dado que es el idóneo para explicar como la arquitectura alcanza las metas de calidad deseadas por el cliente. En particular, durante la etapa en la cual se presenta la arquitectura, la presencia del arquitecto resulta imprescindible, ya que es vital para un buen desarrollo a posteriori de la evaluación que los evaluadores comprendan en detalle la arquitectura propuesta. Ante la falta de esta instancia, estudiamos la arquitectura a partir de la documentación existente. Como la misma no respondió nuestras expectativas (según lo que recomendado por ATAM), se procuro una entrevista con el arquitecto del grupo que desarrolló Graphead. El objetivo de esta entrevista, fue profundizar en el análisis de la arquitectura y comprender como esta satisface los requerimientos de los atributos de calidad.

En la entrevista con el arquitecto se analizó el producto Graphead desde distintos puntos de vista, y se discutió acerca de la utilidad y el manejo de los diferentes COTS utilizados en desarrollo del mismo.

Como ya se mostró en el capítulo tres, la arquitectura propuesta es una arquitectura en cuatro capas. Las razones esgrimidas por el arquitecto para la utilización de esta propuesta fue la siguiente:

- Encapsular en la capa de presentación la lógica de presentación de los casos de uso.
- Utilizar a la capa de servicios del sistema para resolver los casos de uso.
- El objetivo de la capa de servicios del negocio es desacoplar los conceptos de la realidad planteada de la resolución de los casos de uso del sistema, con el fin de que esta capa pudiera ser reutilizada por otros sistemas.
- La capa de infraestructura, posteriormente llamada persistencia, tenía al principio la responsabilidad de contener a los COTS que se utilizaran para el desarrollo. Luego fue diseñada para contener los subsistemas encargados de realizar la persistencia.

El arquitecto considera que la arquitectura que el sugirió era la adecuada para el problema planteado. A su criterio, los inconvenientes que surgieron durante el desarrollo se deben en gran medida a que no se siguió el diseño que realizó. En particular plantea las siguientes apreciaciones:

- Se repite la lógica en tres de las cuatro capas. A nivel de presentación, servicios del negocio y persistencia existe en cada una de ellas un modelo de la realidad. Cuando en principio debería existir un único modelo, en la capa de servicios del negocio, el cual sería utilizado por el resto de las capas. Además de la redundancia de la lógica, y por ende de código, la capa de servicios del sistema se transformó en una simple intermediaria de data types, sin realizar la labor para la cual fue pensada, dado que los casos de uso del sistema también son resueltos por la capa de servicios del negocio. Esto trae consigo que en todo momento de la ejecución del programa, existen en memoria tres instancias completas del modelo (los de las capas de presentación, servicios del negocio y persistencia), y una en almacenamiento secundario (el de la capa de persistencia). Entonces, con cada cambio que se realiza a un proceso, se suceden validaciones y actualizaciones por triplicado (más el acceso a datos para actualizar la imagen del modelo de persistencia en almacenamiento secundario), lo cual afecta negativamente a la performance del sistema.
- Estima que la decisión de actualizar la base de datos en tiempo real no es la adecuada. Según su juicio, esto trae consigo una degradación de la performance, más aun tomando en cuenta que la actualización no esta optimizada, ya que cada vez que se quiere actualizar un atributo, se actualiza toda la base de datos.

- En su opinión la mala performance no solo se debe a lo anteriormente expresado, afirma que la principal causa de dicha mala performance es producto de algoritmos deficientes. Para ejemplificar, el algoritmo de actualización de un proceso, elimina al mismo y lo crea de nuevo con los cambios realizados. Este algoritmo es utilizado en las tres capas que mantienen el modelo del proceso. Por lo tanto, este algoritmo se ejecuta tres veces cada vez que un proceso es modificado.
- Debido a que los tiempos sobre el final del proyecto apremiaron, se tomaron, a su entender, malas decisiones de implementación. Por ejemplo, hubo un abuso del copy paste, derivando en clases de más de 5000 líneas de código, en las cuales cada cierto número de líneas existen bloques de código idéntico, repetido varias veces. No se comento debidamente los algoritmos codificados, resultando en clases únicamente entendibles para la persona que las implementó. Otro efecto del factor tiempo, fue el hecho que algunos subsistemas, en particular los referentes a la persistencia, fueron por necesidad cambiados arbitrariamente de capa. Con lo cual, el objetivo planteado de mantener un bajo acoplamiento entre capas, se desvanece con el desplazamiento de subsistemas con responsabilidades completamente dispares, entre las mismas. Por nombrar un caso, subsistemas encargados de la persistencia fueron colocados en la capa de servicios del negocio. Es más, algunos de estos subsistemas, ni siquiera eran movidos entre capas, si necesitaban interactuar con el usuario, hacían su propia presentación. Consecuentemente, existen subsistemas cuya responsabilidad es la persistencia que interactúan directamente con el usuario, sin utilizar la capa de presentación, ya que dentro de sus clases se encuentran implementadas interfaces gráficas.

Otro punto tratado en esta entrevista fue el análisis de los distintos COTS utilizados en el desarrollo. A continuación detallamos las valoraciones realizadas, sobre este punto, por el arquitecto:

- Para resolver el mapeo de los objetos a las tablas de la base de datos se utilizó, como ya hemos mencionado en el capítulo 3, el framework Hibernate. El arquitecto afirma que esta herramienta no fue utilizada correctamente, y esto se debió principalmente a que no se realizó una debida investigación de la misma. Dado el poco conocimiento que se tenía sobre el uso del framework en cuestión, no se aprovechó todo su potencial, realizando manualmente operaciones que en principio serían resueltas por Hibernate. En si no se le permitió a la herramienta decidir cuando persistir, sino que se hacia cada vez que se le solicitaba. Hibernate para asegurar la performance tiene ciertas precondiciones, al no cumplirse estas, no se puede asegurar como quedará la performance resultante. En particular, como se hacen actualizaciones de la base de datos en tiempo real, se obliga al framework a realizar accesos a la base de datos cada vez que se quiere persistir, sin aprovechar el cache que este implementa, para así disminuir estos accesos y con esto mejorar los tiempos de latencia. Otro aspecto a destacar es, que no solo el uso de Hibernate no es el adecuado para aprovechar sus bondades, sino que además es oscuro. Particularmente, los subsistemas que lo utilizan están repartidos en dos capas arquitectónicas diferentes, y además la codificación de las interacciones de los subsistemas de persistencia con el framework es críptica.
- La opinión del arquitecto acerca del resto de los COTS utilizados (Castor, Xalan, Fop, Jasper y JGraph) es muy buena. Considera que los mismos fueron de gran utilidad para el desarrollo del sistema. En particular destaca Xalan, muy útil para crear el sitio web del proceso, uno de los puntos fuertes del producto. Otro COTS a destacar sobre el resto es JGraph. El mismo fue utilizado para realizar la interfaz gráfica, la cual tuvo una muy buena aceptación por parte del cliente. Sobre el planteo que le hicimos de que Graphead interopere con otro sistema, menciona que el xml generado por la herramienta Castor es un buen modelo de los procesos construidos, el cual podría ser utilizado por cualquier otro sistema.

Consideramos que la entrevista resultó productiva para la evaluación. Nos amplio la visión que teníamos de Graphead, y en particular nos aclaró detalles de la arquitectura y su implementación. Luego de efectuada esta instancia, entendemos que las recomendaciones que realiza ATAM acerca del involucramiento de los stakeholders en la evaluación son apropiadas. Permiten al equipo de evaluación comprender el porqué de cada decisión tomada, mitigando las dudas que surgen sobre el sistema que esta siendo evaluado y por lo tanto mejorando el resultado final de la evaluación.

6 ÁRBOL DE UTILIDAD

Atributo de Calidad	Refinamiento del Atributo	Escenarios
Performance	Tiempo de respuesta	Informar de la aceptación de un alta de actividad en menos de 0.3 segundos **(H,L)**
		Informar de la aceptación de la carga de un proceso dado en menos de 0.3 segundos **(M,L)**
		Informar de la aceptación de un pedido de versionado de proceso en menos de 0.3 segundos **(M,L)**
	Throughput	Generar sitio Web de un proceso con 150 elementos en menos de 3 minutos **(L,M)**
		Generar sitio Web de un proceso con 300 elementos en menos de 10 minutos **(L,M)**
		Versionar un proceso con 150 elementos en menos de 3 segundos **(H,H)**
		Versionar un proceso con 300 elementos en menos de 10 segundos **(H,H)**
	Latencia	Cargar un proceso con 150 elementos en menos 1 minuto **(M,M)**
		Cargar un proceso con 300 elementos en menos 3 minutos **(H,M)**
		Obtener una versión anterior de un proceso con 150 elementos en menos de 1 minuto **(M,M)**
		Obtener una versión anterior de un proceso con 300 elementos en menos de 3 minutos **(M,M)**
		Generar formato exportable de un proceso de 300 elementos en menos de 3 minutos **(M,M)**
		Importar un proceso de 300 elementos en menos de 5 minutos **(M,M)**
		Agregar una actividad a un proceso de 300 elementos en menos de 10 segundos **(M,M)**
	Carga de trabajo	Agregar un nuevo elemento a un proceso de 300 elementos, no aumentará los tiempos de latencia en más de un 15% **(H,H)**

Atributo de Calidad	Refinamiento del Atributo	Escenarios
Maintainability	Cambios en un subsistema no requiere cambios en otros subsistemas	Cambiar el motor de la base de datos implica cambiar un subsistema **(L,M)**
		Cambiar un COTS no requiere cambiar más de un subsistema **(L,M)**
		Cambiar una propiedad del subsistema actividad no implica modificar otros subsistemas **(L,M)**
Modifiability	Cambiar COTS	Cambiar el motor de la base de datos en menos de 10 día/persona **(L,M)**
		Cambiar el generador web en menos de 5 día/persona **(L,M)**
		Cambiar el generador pdf en menos de 5 día/persona **(L,M)**
Reusability	Reusabilidad de componentes	Los conceptos del negocio son utilizables por distintas aplicaciones **(M,M)**

6.1 Lista priorizada de escenarios

Ranking	Escenarios	Observaciones
1	Versionar un proceso con 300 elementos en menos de 10 segundos	Performance
2	Versionar un proceso con 150 elementos en menos de 3 segundos	Performance
3	Agregar un nuevo elemento a un proceso de 300 elementos, no aumentará los tiempos de latencia en más de un 15%	Performance
4	Cargar un proceso con 300 elementos en menos 3 minutos	Performance
5	Informar de la aceptación de un alta de actividad en menos de 0.3 segundos	Performance
6	Los conceptos del negocio son utilizables por distintas aplicaciones	Reusability
7	Agregar una actividad a un proceso de 300 elementos en menos de 10 segundos	Performance
8	Obtener una versión anterior de un proceso con 300 elementos en menos de 3 minutos	Performance
9	Generar formato exportable de un proceso de 300 elementos en menos de 3 minutos	Performance
10	Importar un proceso de 300 elementos en menos de 5 minutos	Performance
11	Cargar un proceso con 150 elementos en menos 1 minuto	Performance
12	Obtener una versión anterior de un proceso con 150 elementos en menos de 1 minuto	Performance
13	Informar de la aceptación de la carga de un proceso dado en menos de 0.3 segundos	Performance
14	Informar de la aceptación de un pedido de versionado de proceso en menos de 0.3 segundos	Performance

Ranking	Escenarios	Observaciones
15	Generar sitio Web de un proceso con 300 elementos en menos de 10 minutos	Performance
16	Generar sitio Web de un proceso con 150 elementos en menos de 3 minutos	Performance
17	Cambiar el motor de la base de datos en menos de 10 día/persona	Modifiability
18	Cambiar el generador web en menos de 5 día/persona	Modifiability
19	Cambiar el generador pdf en menos de 5 día/persona	Modifiability
20	Cambiar el motor de la base de datos implica cambiar un subsistema	Maintainability
21	Cambiar un COTS no requiere cambiar más de un subsistema	Maintainability
22	Cambiar una propiedad del subsistema actividad no implica modificar otros subsistemas	Maintainability

7 ANÁLISIS DE LAS PROPUESTAS ARQUITECTÓNICAS

Basados en los escenarios priorizados que fueron obtenidos en el capitulo anterior, serán analizadas las propuestas arquitectónicas descritas en el capítulo 3. El objetivo de este paso es demostrar, o refutar, si las propuestas arquitectónicas que realizó el arquitecto satisfacen los requerimientos de los atributos de calidad.

7.1 Análisis de Escenarios

7.1.1 Escenario 1

Análisis de una Propuesta Arquitectónica				
Escenario #: 1	**Escenario**	Versionar un proceso con 300 elementos en menos de 10 segundos		
Atributo	Performance – Throughput			
Entorno	Proceso con 300 elementos			
Estímulo	Se invoca el versionado			
Respuesta	Versiona el proceso en menos de 10 segundos			
Decisión Arquitectónica	**Sensitivity**	**Tradeoff**	**Riesgo**	**No riesgo**
Cada subsistema accede directamente a la BD	S1	T1,T2	R1,R2	
Un subsistema que encapsule el acceso a datos (p.e.: uso de Hibernate)	S2	T3		
Razonamiento	S1. Disminuye la confiabilidad. S2. Aumenta la confiabilidad. R1. Aumenta el riesgo que los datos queden inconsistentes ante una eventual falla. R2. Aumento de dependencia entre la aplicación y la BD. T1. Mejora la performance, pero disminuye la maintainability. T2. Mejora la performance, pero disminuye la reusability. T3. Aumenta la reusability, pero empeora la performance.			

7.1.2 Escenario 3

Análisis de una Propuesta Arquitectónica				
Escenario #: 3	**Escenario**	Agregar un nuevo elemento a un proceso de 300 elementos, no aumentará los tiempos de latencia en más de un 15%		
Atributo	Performance – Carga de trabajo			
Entorno	Proceso con 300 elementos			
Estímulo	Se agrega un elemento			
Respuesta	Los tiempos de latencia no aumentan más de un 15%			
Decisión Arquitectónica	**Sensitivity**	**Tradeoff**	**Riesgo**	**No riesgo**
Actualización en tiempo real de la DB		T1		
Mantener un único cache del proceso		T2	R1	
Razonamiento	R1. Aumenta el riesgo que se pierdan datos ante una eventual caída del sistema. T1. Aumenta la confiabilidad, pero empeora la performance. T2. Mejora la performance, pero disminuye la confiabilidad.			

7.1.3 Escenario 4

Análisis de una Propuesta Arquitectónica					
Escenario #: 4	**Escenario**	Cargar un proceso con 300 elementos en menos 3 minutos			
Atributo	Performance – Latencia				
Entorno	Proceso con 300 elementos				
Estímulo	Se invoca la carga del proceso				
Respuesta	Carga el proceso en menos de 3 minutos				
Decisión Arquitectónica		**Sensitivity**	**Tradeoff**	**Riesgo**	**No riesgo**
Carga a demanda		S1			
Cargar todo el proceso		S2			
Razonamiento	S1. Mejora la performance al instanciar solo lo necesario. S2. Instanciar es una de las operaciones más costosas, por lo tanto traer todo un proceso a memoria puede resultar en una opción poco performante.				

7.1.4 Escenario 5

Análisis de una Propuesta Arquitectónica					
Escenario #: 5	**Escenario**	Informar de la aceptación de un alta de actividad en menos de 0.3 segundos			
Atributo	Performance – Tiempo de respuesta				
Entorno	Durante una ejecución normal				
Estímulo	Se invoca un alta de actividad				
Respuesta	El sistema informa que se esta procesando el pedido en menos de 0.3 segundos				
Decisión Arquitectónica		**Sensitivity**	**Tradeoff**	**Riesgo**	**No riesgo**
Manejar un evento de aviso				R1	
Orientación a Aspectos		S1,S2			
Razonamiento	S1. Aumenta la maintainability. S2. Favorece la modifiability. R1. Puede no resultar sencillo disparar y manejar eventos.				

7.1.5 Escenario 6

Análisis de una Propuesta Arquitectónica				
Escenario #: 6	**Escenario** Los conceptos del negocio son utilizables por distintas aplicaciones			
Atributo	Reusability			
Entorno	Ambiente de diseño e implementación			
Estímulo	Otra aplicación desea utilizar los conceptos del negocio			
Respuesta	La nueva aplicación utiliza los conceptos del negocio			
Decisión Arquitectónica	**Sensitivity**	**Tradeoff**	**Riesgo**	**No riesgo**
Orientación a Componentes	S1,S2,S3		R1	
Capas Jerárquicas	S2,S3	T1	R1	
Orientación a Servicios	S1,S2,S3		R1,R2	
Razonamiento	S1. Mejora la reusability. S2. Aumenta la maintainability. S3. Favorece la modifiability. R1. No siempre son claras las particiones del sistema el sistema. R2. Agrega complejidad a las etapas de diseño e implementación. T1. Mejora la reusability, pero se puede ver afectada la performance del sistema, eso depende de la cantidad de capas y de si es un modelo estricto o relajado.			

7.2 Observaciones

Las propuestas arquitectónicas analizadas intentan satisfacer los escenarios que describen los requerimientos de los atributos de calidad. No todas las propuestas analizadas fueron realizadas por el arquitecto, sino que la mayoría fueron elaboradas por el equipo de evaluación. Esto se debió a que del diseño encontrado en la documentación y la información recavada durante la entrevista, no se halló la forma en que las propuestas arquitectónicas planteadas por el arquitecto alcanzaban los requerimientos de los atributos de calidad.

8 CONCLUSIONES

Este capítulo tiene como objetivo concluir en dos aspectos de la evaluación realizada. El primero de ellos es razonar acerca de los resultados obtenidos de la evaluación de la arquitectura de software del producto Graphead, y el segundo analizar la aplicación particular efectuada de ATAM para llevar a cabo la evaluación mencionada.

8.1 Graphead

Cabe recordar que a pesar de que el equipo de evaluación esta sistemáticamente trabajando en intentar de entender las propuestas arquitectónicas, es inevitable que, a veces, haga recomendaciones de como la arquitectura podría haber sido diseñada, o analizada diferente. Estas estrategias de mitigación podrían estar relacionadas con el proceso, podrían ser directivas, o podrían ser técnicas. Sin embargo, no es una parte integral del ATAM ofrecer estrategias de mitigación. El objetivo del ATAM es localizar los riesgos de la arquitectura.

A partir de los requerimientos de los atributos de calidad destacados por el cliente se fijaron las metas de la evaluación. Se estudió el producto a través de la documentación buscando entender la forma en que la arquitectura cumplía con las expectativas del cliente. Para realizar este estudio se generaron veintidós escenarios distintos que modelan los requerimientos de los atributos de calidad especificados por el cliente. Cada uno de estos escenarios fue ubicado en una hoja del árbol de utilidad, según el atributo de calidad que estuviera poniendo a prueba. Los escenarios generados fueron ponderados y posteriormente priorizados. Una vez ordenados por la prioridad elegida, del total de los mismos se seleccionaron los siguientes:

Ranking	Escenarios	Observaciones
1	Versionar un proceso con 300 elementos en menos de 10 segundos	Performance
3	Agregar un nuevo elemento a un proceso de 300 elementos, no aumentará los tiempos de latencia en más de un 15%	Performance
4	Cargar un proceso con 300 elementos en menos 3 minutos	Performance
5	Informar de la aceptación de un alta de actividad en menos de 0.3 segundos	Performance
6	Los conceptos del negocio son utilizables por distintas aplicaciones	Reusability

Se realizó un análisis para cada uno de estos escenarios priorizados, buscando encontrar la propuesta arquitectónica más adecuada para satisfacer el requerimiento del atributo de calidad específico. En base a estas propuestas el arquitecto del equipo que itere el desarrollo de Graphead debe analizar los sensitivity points, tradeoff points y riesgos que se detectaron de cada una. Una vez considerado el impacto que cada propuesta pueda tener sobre la arquitectura de software, este debe seleccionar las que considere más convenientes para alcanzar los requerimientos de los atributos calidad. Es importante que al seleccionar una propuesta arquitectónica se consideren todos los aspectos de la evaluación, lo que significa que guiarse únicamente por los sensitivity points, sin tomar resguardos de los riesgos que cada decisión trae consigo pone en peligro el éxito del proyecto. Es por este motivo que ATAM incluye en sus salidas, además de las propuestas arquitectónicas, la lista de sensitivity points, tradeoff points, riesgos y no riesgos (que en esta evaluación no surgieron).

Sería conveniente que no se perdieran de vista los detalles acerca de la implementación. La evaluación no solo produjo recomendaciones sobre como diseñar la

arquitectura, sino que también trajo a la luz problemas en otras áreas. Se detectaron incompatibilidades entre el diseño y la implementación. En particular, a pesar que no es el cometido del método, se constató que la lógica del sistema no prevé la satisfacción de requerimiento de calidad alguno.

8.2 ATAM

El equipo de evaluación procuró en todo momento seguir los lineamientos planteados por ATAM durante el desarrollo de la evaluación. Sin embargo, dadas las particularidades que envolvían al producto a evaluar, algunos pasos del método fueron adaptados a la realidad enfrentada y otros eliminados.

Dentro de las peculiaridades que se afrontaron, una a destacar es que el único representante de los stakeholders que formó parte de la evaluación fue el cliente. Esto se debió a las características del equipo de desarrollo de Graphead. Este fue un grupo de Proyecto de Ingeniería de Software 2004, que por tanto, una vez finalizado el curso, los mismos se desligaron del sistema implementado. En consecuencia, una de las bondades que se destaca de ATAM con mayor énfasis como ser el hecho de que logra juntar a todos los stakeholders en una misma habitación junto al equipo de evaluación fue desaprovechada. Subsiguientemente, se perdió el feedback entre los stakeholders y el equipo de evaluación, lo cual tuvo un corolario negativo en varios aspectos de la evaluación realizada.

Por lo antes descrito, la presentación del método solo se realizó para el cliente, con el objetivo que este entendiera la tarea que se iba a realizar y la importancia de su participación en distintas instancias durante la evaluación.

Una de las instancias en la que la participación del cliente es clave es la presentación de las pautas del negocio. Durante la misma, el cliente explicó finalidad que tiene el producto, y los aspectos de Graphead que no le satisficieron. Del mismo modo, transmitió cuales eran sus expectativas de la evaluación, y también como esperaba que esta ayudara a mejorar la calidad del producto en la próxima iteración de mismo. Fue en esta reunión donde el equipo de evaluación obtuvo los requerimientos de los atributos de calidad que el tenía del sistema, los cuales guiarían el resto de la evaluación.

Una vez conocidas las pautas del negocio, el equipo de evaluación debió enfrentarse a la arquitectura de software del sistema por si solo, dado que como ya fue mencionado no se contaba con la presencia del arquitecto. Entonces, para poder identificar y entender las propuestas arquitectónicas realizadas por el arquitecto se recurrió a la documentación existente. En la misma se encontraron varias falencias, las cuales dificultaban entender de que manera la arquitectura diseñada satisfacía los requerimientos de los atributos de calidad. En vista de la presente dificultad para comprender en detalle las propuestas arquitectónicas descritas en la documentación, el equipo de evaluación gestionó una entrevista con el arquitecto. En la misma se obtuvo valiosa información acerca de Graphead. Consideramos que esta entrevista tuvo una influencia positiva en el resultado de la evaluación, y confirma porque para ATAM es tan importante la participación de los distintos stakeholders, en particular el arquitecto, en la evaluación. Creemos que sin una guía de las personas involucradas con el sistema, la evaluación puede llegar a producir resultados que no se corresponden con la realidad.

Dado que la documentación existente no satisfacía al equipo de evaluación, este (como ATAM recomienda) debió detener la evaluación hasta que dicha documentación fuera perfeccionada. Con esto el método ya comienza a mejorar la calidad total del producto desarrollado. Esta fue otra de las bondades que se desaprovecharon de ATAM, dado que ante la falta del equipo de desarrollo, la evaluación continuó. Una vez más vemos la importancia de la interacción de los evaluadores con los stakeholders, esta vez en particular con el equipo de desarrollo.

En base a los requerimientos de los atributos de calidad obtenidos en la entrevista con el cliente, se generó el árbol de utilidad. Este contiene los escenarios que ponen a prueba a las propuestas arquitectónicas, para ver si estas alcanzan los requerimientos de los atributos de calidad especificados. Los escenarios producidos fueron priorizados por el equipo de evaluación, según los lineamientos marcados por el cliente.

Se analizaron diferentes propuestas arquitectónicas para los escenarios priorizados. Estas propuestas, al no contar con el arquitecto, fueron en su mayoría realizadas por los evaluadores, a las que se le sumaron las que se obtuvieron de la documentación. En el análisis de cada propuesta se buscó determinar sobre que atributos de calidad influía y que riesgos o no riesgos introducía.

Debido a la realidad a la cual se enfrentaba el equipo de evaluación, se omitió la instancia en la cual todos los stakeholders proponen escenarios, y luego los analizan junto con los evaluadores. ATAM considera que a pesar de que los evaluadores a esta altura llevan un buen tiempo trabajando sobre la arquitectura y la conocen en detalle, siempre quedan algunas características ocultas, las cuales pueden llegar a ser críticas para alcanzar los requerimientos de los atributos de calidad. Es por este motivo que se introduce este paso. Por lo tanto, al no realizarlo, se perdió la oportunidad de descubrir nuevos sensitivity points, tradeoff points, riesgos y no riesgos.

Como resultado de la aplicación del método se obtuvo una lista de escenarios priorizados, junto con las diferentes propuestas arquitectónicas que los intentan satisfacer. A partir de estas propuestas arquitectónicas se descubrieron sensitivity points, tradeoff points y riesgos. Las propuestas arquitectónicas relevadas son una excelente guía para tomar futuras decisiones arquitectónicas, puesto que se analizó su posible impactó en la arquitectura y atacan directamente las expectativas de calidad que el cliente tiene del sistema.

Consideramos que el método resultó muy útil para evaluar la arquitectura de software, a pesar de que no se siguió la metodología propuesta en forma estricta, incluso aún omitiendo pasos. La evaluación permitió un enfoque amplio del sistema, en lugar de uno estrecho o con preocupaciones a corto plazo. Gracias a este tipo de enfoque que plantea el método de evaluación se descubrieron riesgos ocultos y errores explican los problemas que tiene el producto. Creemos que ATAM se puede adaptar a las necesidades particulares de cada negocio.

9 GLOSARIO

- **Actividad:**
  Parte de un proyecto que tiene lugar durante un periodo de tiempo.
  Una actividad es algo que un rol hace y eso proporciona un resultado significativo en el contexto del proyecto.
  Las actividades están estrechamente relacionadas a los entregables. Los entregables son la entrada y salida de las actividades, y el mecanismo mediante el cual se comunica la información entre las actividades.

- **Hito:**
  Marca la culminación de cada una de las fases de un ciclo de desarrollo de un proyecto.

- **Proceso:**
  Conjunto ordenado de tareas (o serie de pasos) que involucran actividades, restricciones y recursos y que producen una determinada salida.
  Impone consistencia y estructura sobre un conjunto de actividades, que permite se pueda repetir de igual modo.
  Forma de transferir experiencias.

- **Definición:**
  Explicación exacta y clara de un concepto. Expone con precisión la naturaleza de lo que se define.

- **Rol:**
  Un rol es una definición abstracta de un conjunto de actividades y los artefactos producidos. Puede ser realizado por una persona o un grupo de personas trabajando juntas como un equipo. Un miembro del equipo de proyecto generalmente cumple varios roles y se especifica que comportamiento y responsabilidades debe cumplir dicho miembro.

- **Entregable:**
  Un entregable es todo tipo de información creada, producida, cambiada o usada por el proceso.
  Un entregable es un producto del proceso: las personas de los distintos roles usan entregables para realizar actividades, y producen entregables en la realización de las actividades. Aunque una persona puede ser la "dueña" o responsable del entregable, otras personas pueden usarlo, incluso actualizarlo si se les da el permiso correspondiente.

- **Memoria Organizacional:**
  Este término, utilizado en el contexto del producto a desarrollar, es el historial de proyectos de Ingeniería de Software realizado por los estudiantes de dicho curso.

- **Agenda:**
  Forma por la cual se organizan las actividades en el tiempo.
  Marcando un orden parcial o total definido por las dependencias entre los entregables que son salida o entrada de una actividad.

- **Fase:**
  Forma de organizar un proceso. Cada fase de un proceso tiene un objetivo propio marcado. Las fases tienen un orden total, lo que refiere a que una fase no puede comenzar hasta que la anterior no esté finalizada.

- **Iteración:**
  Es la forma de particionar la duración de una fase del proceso de desarrollo.
  En dicha partición, se repite un subconjunto de las actividades de un proceso (estas pueden ser diferentes o pueden repetirse en distintas iteraciones).

- **Área:**
  Subgrupo de actividades de un proceso con un objetivo común particular, pero asociado con el objetivo del proceso mismo. Un área además puede tener roles específicos. Una persona puede cumplir un rol para un área y otro rol distinto para otra área del mismo proceso.

- **Tiempo de respuesta:**
  Tiempo en que la aplicación informa al usuario que su pedido esta siendo procesado, o que se retorna el control al solicitante.

- **Throughput:**
  La velocidad con que una aplicación procesa los datos.

- **Latencia:**
  El tiempo que transcurre entre un estímulo y su respuesta.

- **Carga de trabajo:**
  La cantidad de trabajo que una aplicación realiza, o se espera que realice, en un período de tiempo específico.

10 REFERENCIAS

- Especificación de Requerimientos v2.2; grupo 3 PIS 2004.
- Requerimientos Suplementarios v1.2; grupo 3 PIS 2004.
- Modelo de Casos de Uso v4.0; grupo 3 PIS 2004.
- Modelo de Dominio v1.3; grupo 3 PIS 2004.
- Modelo de Diseño v1.6; grupo 3 PIS 2004.
- SAD v1.2; grupo 3 PIS 2004.
- Glosario v1.0; grupo 3 PIS 2004.
- ATAM v3.0; Alberto Castro y Martín Germán; Pasantía: Evaluación de Arquitecturas de Software.
- Evaluating Software Architectures: Methods and Case Studies; Paul Clements, Rick Kazman y Mark Klein; Marzo 2004; ISBN 0-201-70482-X.
- Using the Architecture Tradeoff Analysis Method to Evaluate a Wargame Simulation System: A Case Study; Lawrence G. Jones y Anthony J. Lattanze; Diciembre 2001.
- Evaluación Arquitectónica de un Software Basado en Componentes; Maria Pérez, Luís Mendoza, Dakar Parada y George Di Paula; 2004.
- New Software Performance AntiPatterns: More Ways to Shoot Yourself in the Foot; Connie U. Smith; Lloyd G. Williams; 2002.
- Hibernate Reference Documentation v3.0.5.
- www.fing.edu.uy/inco - 03/10/2005
- www.fing.edu.uy/inco/grupos/gris - 03/10/2005
- www.hibernate.org - 03/10/2005
- www.answers.com - 03/10/2005