A lo largo de este capítulo se tratará el método utilizado en la elaboración de este proyecto y las consideraciones que se han llevado a cabo a lo largo de la elaboración del mismo. Así mismo se explicará la planificación realizada.

10.1. Metodología

En la primera fase de desarrollo del proyecto iniciamos un proceso de aprendizaje y cuantificación de dificultad de objetivos con el fin de poder hacer una correcta planificación. Durante las primeras semanas de la elaboración de este, el principal objetivo fue la investigación sobre MDA, la herramienta Rational Software Architect, y las transformaciones en este. Llevamos a cabo gran parte de los tutoriales incluidos en la herramienta para familiarizarnos con el entorno.

La siguiente fase fue la del acondicionamiento de nuestras máquinas para poder trabajar de manera adecuada: aumento de memoria RAM, puesto que la RSA necesita bastantes recursos, instalación del software pertinente y preparación de un servidor CVS.

Para maximizar nuestras garantías de éxito nos propusimos seguir algunas de las directrices de la metodología RUP. Es por ello que a lo largo de esta sección, nos referiremos a varios de los elementos que se deben incluir en la documentación técnica de un proyecto que implanta RUP.

10.1.1. Alcance del proyecto

Una parte muy importante a la hora de llevar a cabo un proyecto es definir el alcance de este. Determinar las fronteras del proyecto es esencial para cumplir plazos y saber responder adecuadamente a la pregunta *qué es el proyecto y qué no*. En definitiva llegar al éxito del proyecto.
Gran cantidad de proyectos, en especial de software, han visto cómo acababan en *papel mojado*\(^1\) por no llevar una correcta planificación debido a que no se había establecido hasta dónde se quería llegar. Para no cometer este error o en su defecto reducir el riesgo de no terminar a tiempo, se determinó cuáles serían los principales objetivos que se deseaban obtener con el proyecto.

Para ilustrar los límites del proyecto, en el diagrama de la figura 10.1.1 podemos ver cada una de las partes que se han estudiado y desarrollado.

![Diagrama de límites del proyecto](image)

**Figura 10.1.1:** Límites del proyecto

El proyecto está organizado de manera que se pueda actuar de forma iterativa: su organización permite ampliar tanto como se desee/pueda los casos de estudio de desarrollo MDA, en función de la dificultad a la hora de llevar a cabo cada una de las etapas.

El objetivo es marcar metas tan pronto como la incertidumbre se minimice tanto como para acotar fechas. No olvidemos que este es un proyecto de investigación en el que se estudia la viabilidad de adaptación de RSA a la metodología MDA.

Como se puede observar en la figura 10.1.1, se puede avanzar la investigación en dos dimensiones: por una parte se puede fijar un caso de estudio y maximizar el número de tecnologías que se pueden llegar a generar a partir de los mismos modelos CIM y PIM. La segunda dimensión de investigación consiste en aumentar el número de modelos implementados en Rational Software Architect para trabajar con MDD.

---

\(^1\)Se estima que el 31% de los proyectos de desarrollo de software son cancelados durante su desarrollo
10.1. METODOLOGÍA

Decisión:
Nuestro principal objetivo ha sido aumentar el número de casos de estudio, pues-
to que la cantidad de trabajo que repercute en sendos proyectos es equiparable.
En el caso de aumentar el número de tecnologías que genera un caso de estudio
particular influye de manera más significativa en el proyecto de más bajo nivel.

10.1.2. Enfoque del proyecto

A la hora de trabajar en este proyecto de software, se han querido analizar aspectos no
menos importantes que el propio desarrollo como el Project Management. Para explicar el
enfoque que ha tomado el proyecto se ha dividido en tres importantes ámbitos, que son los
siguientes:

- Ingeniería del Software
- Project Management
- Programación

La siguiente figura ilustra perfectamente los ámbitos de desarrollo del proyecto, y con
color el énfasis/importancia que ha tomado.

![Diagrama de Triángulo del enfoque]

Figura 10.1.2: Triángulo del enfoque

Cómo se puede apreciar en la figura el trabajo realizado se ha enfocado principalmente
da aspectos de ingeniería del software, seguidos de los aspectos de Project Management y
finalmente de los tecnológicos y de programación.
Capítulo 10

Metodología y Planificación

A lo largo de este capítulo se tratará el método utilizado en la elaboración de este proyecto y las consideraciones que se han llevado a cabo a lo largo de la elaboración del mismo. Así mismo se explicará la planificación realizada.

10.1. Metodología

En la primera fase de desarrollo del proyecto iniciamos un proceso de aprendizaje y cuantificación de dificultad de objetivos con el fin de poder hacer una correcta planificación. Durante las primeras semanas de la elaboración de este, el principal objetivo fue la investigación sobre MDA, la herramienta Rational Software Architect, y las transformaciones en este. Llevamos a cabo gran parte de los tutoriales incluidos en la herramienta para familiarizarnos con el entorno.

La siguiente fase fue la del acondicionamiento de nuestras máquinas para poder trabajar de manera adecuada: aumento de memoria RAM, puesto que la RSA necesita bastantes recursos, instalación del software pertinente y preparación de un servidor CVS.

Para maximizar nuestras garantías de éxito nos propusimos seguir algunas de las directrices de la metodología RUP. Es por ello que a lo largo de esta sección, nos referiremos a varios de los elementos que se deben incluir en la documentación técnica de un proyecto que implanta RUP.

10.1.1. Alcance del proyecto

Una parte muy importante a la hora de llevar a cabo un proyecto es definir el alcance de este. Determinar las fronteras del proyecto es esencial para cumplir plazos y saber responder adecuadamente a la pregunta qué es el proyecto y qué no. En definitiva llegar al éxito del proyecto.
Gran cantidad de proyectos, en especial de software, han visto cómo acababan en papel mojado por no llevar una correcta planificación debido a que no se había establecido hasta dónde se quería llegar. Para no cometer este error o en su defecto reducir el riesgo de no terminar a tiempo, se determinó cuáles serían los principales objetivos que se deseaban obtener con el proyecto.

Para ilustrar los límites del proyecto, en el diagrama de la figura 10.1.1 podemos ver cada una de las partes que se han estudiado y desarrollado.

![Diagrama de ejemplo]

**Figura 10.1.1:** Límites del proyecto

El proyecto está organizado de manera que se pueda actuar de forma iterativa: su organización permite ampliar tanto como se desee/pueda los casos de estudio de desarrollo MDA, en función de la dificultad a la hora de llevar a cabo cada una de las etapas.

El objetivo es marcar metas tan pronto como la incertidumbre se minimice tanto como para acotar fechas. No olvidemos que este es un proyecto de investigación en el que se estudia la viabilidad de adaptación de RSA a la metodología MDA.

Como se puede observar en la figura 10.1.1, se puede avanzar la investigación en dos dimensiones: por una parte se puede fijar un caso de estudio y maximizar el número de tecnologías que se pueden llegar a generar a partir de los mismos modelos CIM y PIM. La segunda dimensión de investigación consiste en aumentar el número de modelos implementados en Rational Software Architect para trabajar con MDD.

---

1Se estima que el 31% de los proyectos de desarrollo de software son cancelados durante su desarrollo.
10.1. METODOLOGÍA

Decisión:
Nuestro principal objetivo ha sido aumentar el número de casos de estudio, puesto que la cantidad de trabajo que repercute en sendos proyectos es equiparable. En el caso de aumentar el número de tecnologías que genera un caso de estudio particular influye de manera más significativa en el proyecto de más bajo nivel.

10.1.2. Enfoque del proyecto

A la hora de trabajar en este proyecto de software, se han querido analizar aspectos no menos importantes que el propio desarrollo como el Project Management. Para explicar el enfoque que ha tomado el proyecto se ha dividido en tres importantes ámbitos, que son los siguientes:

- Ingeniería del Software
- Project Management
- Programación

La siguiente figura ilustra perfectamente los ámbitos de desarrollo del proyecto, y con color el énfasis/importancia que ha tomado.

![Triángulo del enfoque](image)

Figura 10.1.2: Triángulo del enfoque

Cómo se puede apreciar en la figura el trabajo realizado se ha enfocado principalmente a aspectos de ingeniería del software, seguidos de los aspectos de Project Management y finalmente de los tecnológicos y de programación.
10.1.2.1. Ingeniería del Software

Ha sido el enfoque que ha tomado más fuerza en el proyecto. Con este enfoque ganamos un aumento de la productividad y la calidad del software desarrollado.

La Ingeniería del Software incluye los procesos de análisis de requisitos, especificación, diseño, programación, prueba, documentación y mantenimiento. Puesto que el trabajo que hemos realizado es una prueba piloto de un producto mucho mayor, hemos creído interesante no centramos en el resultado final en beneficio al estudio del método en sí.

Los beneficios de seguir este procedimiento de desarrollo de software son muchos. Entre ellos hay una mejor eficiencia, flexibilidad, portabilidad, fiabilidad, mantenibilidad del software. Éstos requisitos implican otro requisitos de tipo no funcional cómo por ejemplo el económico.

El objetivo final de la Ingeniería del Software es obtener un producto capaz de satisfacer al usuario final, adaptándose a los requisitos impuestos por este.

10.1.2.2. Project Management

Ha sido otro de los pilares fundamentales de nuestro trabajo. La importancia de llevar una gestión correcta del proyecto ha sido fundamental.

Ha habido un conjunto de recursos a gestionar que sin una buena gestión desde el principio hubiera provocado el fracaso del proyecto. La siguiente lista muestra recursos a gestionar dentro del proyecto:

- Planificación de tareas - Gestión del tiempo
- Gestión y organización de recursos - Gestión de recursos
- Adquisición de material
- Control y análisis de progreso

Los dos más importantes han sido la gestión del tiempo y de la gente. Cómo veremos más adelante en el listado de recursos y por comentarios anteriores, el proyecto se ha llevado a cabo por Daniel Albert y Ferran Basora. El hecho de trabajar conjuntamente ha desembocado en la necesidad de una cierta organización en todos los aspectos, que se ha resuelto de la siguiente manera:

- Reuniones periódicas: para compartir impresiones, tomar decisiones, etc.
- Software de control de versiones: para facilitar el control del trabajo concurrente por parte de los integrantes.
10.1. METODOLOGÍA

- **Intercambio de documentación**: creación de un listado con toda la bibliografía interesante para los proyectistas, para recoger conceptos, ideas, etc.

- **Registro de horas de trabajo**: creación de un calendario con la cantidad de horas trabajadas diariamente para posteriormente determinar el desfase de la previsión temporal y la realidad.

La planificación de tareas y gestión del tiempo no ha sido tarea fácil ya que se han tenido que combinar incompatibilidades por parte de los dos proyectistas, aspecto que ha supuesto un valor añadido a la gestión del proyecto.

Otro aspecto que tuvo relevancia en la primera etapa del proyecto, fue la adquisición de material a nivel de programas y hardware para llevar a cabo un desarrollo fluido durante todo el proyecto.

10.1.2.3. **Programación**

La programación ha sido una fase fundamental en el transcurso del trabajo realizado. Sin ella no se podría responder de manera concreta sobre la viabilidad de adaptación de RSA a MDA.

De todos modos, es importante comentar que no es la parte más importante del proyecto. El hecho de que una de las partes del trabajo sea un producto tangible únicamente aporta estabilidad y seguridad al estudio que estamos realizando. Forma parte de una demostración de que nuestra respuesta a la pregunta de si se puede adaptar RSA para que soporte MDA es afirmativa, pero no es lo único que proporcionamos.

El verdadero trabajo realizado es la información en sí, el conocimiento de cómo se debe realizar dicha adaptación.

10.1.3. **Metodología de desarrollo**

10.1.3.1. **Proceso iterativo**

Puesto que el grado de incertidumbre sobre la dificultad en el desarrollo de transformaciones ha sido alto en los inicios de la elaboración de este proyecto, el hecho de llevar a cabo un proceso iterativo supone una garantía de éxito.

El desarrollo mediante proceso iterativo consiste en repetir tantas veces como sea necesario una secuencia de acciones previamente definidas. Una vez completadas todas y cada una de ellas, se vuelve a tomar la primera de las acciones y se vuelven a ejecutar en secuencia. Actuando de esta manera conseguimos minimizar la incertidumbre en cada iteración. En cada vuelta del proceso, los objetivos son más concretos y cualquier necesidad de cambio impacta mínimamente en el proyecto global.
Su principal competidor es el desarrollo en cascada. El desarrollo en cascada pretende enumerar una secuencia de acciones. No se permite comenzar la etapa $i+1$ hasta que no se ha completado la etapa $i$. El problema del desarrollo en cascada es que si en la quinta etapa se detecta un error cometido en la etapa segunda, implica realizar cambios en todas las etapas intermedias. Es por ello que cualquier fallo impacta sobre el tiempo de desarrollo global.

La mayoría de metodologías de trabajo en el mundo del desarrollo software recomiendan el proceso iterativo. Las principales fases de todo proceso iterativo incluyen:

- Planificación
- Análisis de requerimientos
- Análisis y diseño
- Implementación
- Deployment
- Pruebas
- Cierre y Evaluación

### 10.1.3.2. Milestones

Todo proceso iterativo debe terminar en alguna especie de análisis de lo que ha sucedido. Si nos fijamos en aquello que ha pasado, identificamos las riquezas y pobrezas de la etapa, podremos planificar una siguiente vuelta mejor. En nuestras reuniones periódicas, a este proceso le hemos denominado *milestones*.

Al final de cada iteración hemos considerado oportuno listar todo aquello que ha sucedido. Identificar qué puntos nos habían parecido sencillos de solucionar pero que a la hora de la verdad no han resultado como esperábamos. Identificar aquellos elementos que finalmente han resultado más sencillos de lo esperado. Planificar la siguiente etapa mejor, con el fin de ajustar mejor la planificación. Tratar los posibles riesgos materializados y ejecutar su plan de contingencia.

En la metodología RUP este elemento consta como artefacto y se le denomina *Iteration Assessment*.

### 10.1.3.3. Bottom-up

En la elaboración de este proyecto se ha llevado a cabo un proceso de desarrollo *bottom-up* con tal de minimizar riesgos.
10.1. METODOLOGÍA

¿En qué consiste el desarrollo bottom-up? El proceso que es necesario para desarrollar una transformación es bastante complejo, y al inicio de cada transformación resulta un problema difícil de abordar. No obstante, lo que se tiene claro al inicio del proceso es cuál tiene que ser el resultado.

Si nos centramos en la metodología MDA, a la hora de desarrollar un proceso asistido MDD, necesitamos definir cómo deben ser los modelos de los niveles CIM, PIM y PSM para obtener como resultado un código concreto.

Es en esta etapa en la que el nivel de incertidumbre de cómo debe ser la definición de modelos en cada nivel aumenta a medida que el nivel es más abstracto.

![Diagrama de niveles CIM, PIM, PSM y proceso bottom-up](image)

Figura 10.1.3: Proceso bottom-up

A medida que un modelo en un determinado nivel está definido con mayor concreción, el nivel de incertidumbre del modelo inmediatamente superior desciende. Gracias a esta característica, el hecho de definir los modelos de abajo a arriba ayuda a disminuir la incertidumbre y la definición del modelo es más sencilla, por lo que el proceso se optimiza.

10.1.4. Iteraciones

A continuación vamos a explicar las fases por las cuales ha pasado este proyecto y las decisiones más importantes que se han tomado en cada una de ellas. Para cada una de las fases destacaremos aspectos como tiempo, recursos y esfuerzo dedicado.

En primer término transcurre una fase inicial, en la que se hace un estudio de la viabilidad del proyecto. Dentro de este estudio se analizan aspectos como el alcance y repercusión entre otros. En esta fase inicial se establecen parámetros como la metodología a seguir.

Además la fase incluye el estudio de MDA, instalación de herramientas y utilidades necesarias para llevar a cabo el proyecto, etc.

Por otra parte, en la fase inicial se hace una primera estimación de la planificación a seguir. Dicha planificación será replanteada en posteriores fases como consecuencia de
los respectivos *milestones* (comentados en el apartado anterior).

Llegado el punto en que los procesos especificados terminan, entramos en un proceso iterativo en el que se desarrollan los patrones MDA. Cada iteración está compuesta por dos subetapas que llamaremos *Elaboración* y *Construcción*.

- **Elaboración**: en esta subfase se contempla el análisis, diseño y estudio en profundidad del patrón MDA que queremos construir.

- **Construcción**: esta subfase está compuesta por la implementación del patrón MDA.

En este proyecto, se han desarrollado dos patrones MDA que corresponden al CRUD y a Login. Cada uno de estos patrones está incluido dentro de una iteración con sus respectivas subfases de *Elaboración* y *Construcción*.

Para dejar más claro las fases transcurridas, lo expresaremos mediante la figura 10.1.4.

![Figura 10.1.4: Iteraciones](image)

En dicha figura podemos ver gráficamente los esfuerzos que se han dedicado en cada fase.

Finalmente, tal y como se explica en el apartado de metodología de desarrollado (consultar 10.1.3), al final de cada iteración marcamos un *milestone* para hacer una evaluación con el objetivo de encontrar deficiencias en el desarrollo. Gracias a estos perfeccionaremos el proceso de desarrollo.

En la tabla 10.1.1, podemos observar los detalles más destacables de cada fase.

### 10.1.5. Recursos

Los miembros que han intervenido en la elaboración del trabajo son:
10.1. METODOLOGÍA

<table>
<thead>
<tr>
<th>Etapa</th>
<th>Tarea</th>
<th>Fecha Inicio</th>
<th>Fecha Fin</th>
<th>Observaciones</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inicial</td>
<td>Viabilidad</td>
<td>15 Julio</td>
<td>6 Octubre</td>
<td>Estudio de la viabilidad de todo el proyecto</td>
</tr>
<tr>
<td></td>
<td>Planificación</td>
<td>15 Julio</td>
<td>6 Octubre</td>
<td>Definición de la planificación a seguir</td>
</tr>
<tr>
<td></td>
<td>Herramientas</td>
<td>15 Julio</td>
<td>6 Octubre</td>
<td>Instalación de todas las herramientas necesarias para desarrollar el proyecto</td>
</tr>
<tr>
<td></td>
<td>milestone</td>
<td></td>
<td></td>
<td>Tomada la decisión de hacer el CRUD en varias tecnologías, por ejemplo en PHP</td>
</tr>
<tr>
<td>Iteración 1</td>
<td>Elaboración</td>
<td>7 Octubre</td>
<td>18 Noviembre</td>
<td>Análisis y diseño del patrón MDA de CRUD</td>
</tr>
<tr>
<td></td>
<td>Construcción</td>
<td>19 Noviembre</td>
<td>12 Diciembre</td>
<td>Implementación del patrón CRUD</td>
</tr>
<tr>
<td></td>
<td>milestone</td>
<td></td>
<td></td>
<td>Descartada la implementación PHP de CRUD. Tomada la decisión de crear una nuevo patrón MDA, en concreto el Login</td>
</tr>
<tr>
<td>Iteración 2</td>
<td>Elaboración</td>
<td>13 Diciembre</td>
<td>1 Enero</td>
<td>Análisis y diseño del patrón MDA de Login</td>
</tr>
<tr>
<td></td>
<td>Construcción</td>
<td>2 Enero</td>
<td>20 Enero</td>
<td>Implementación del patrón Login</td>
</tr>
<tr>
<td></td>
<td>milestone</td>
<td></td>
<td></td>
<td>Evaluación final del proyecto</td>
</tr>
</tbody>
</table>

Cuadro 10.1.1: Detalles de las fases de desarrollo

- Ernest Teniente López en calidad de Director del proyecto
- Álex Ballarín Latre como contacto externo en IBM
- Daniel Albert Sánchez en calidad de proyectista
- Ferran Basora Roca en calidad de proyectista

Los dos miembros proyectistas se encargarán de decidir todas las decisiones que se deben tomar a lo largo del proyecto, con el consejo del director y de Álex.

Entre ambos miembros se determina qué transformaciones son susceptibles de definición e implementación y de llevar a cabo los *iteration assessment* conjuntamente.

Daniel se encarga de definir, diseñar e implementar todos los elementos que conforman la transformación de un modelo que representa un nivel CIM hasta otro que modeliza un nivel PSM.

Ferran se encarga de definir, diseñar e implementar todos los elementos que conforman la transformación de un modelo que representa un nivel PSM hasta la implementación de dicho modelo en la tecnología que corresponde.
Decisión:
El reparto de cada una de las dos partes del proyecto conjunto se llevó a cabo los primeros días y se tuvieron en cuenta cuestiones como conocimientos, habilidades, inquietudes de cada uno de proyectistas.

10.1.6. Gestión de riesgos

Describimos a continuación los riesgos que contemplamos y las posibles soluciones que plantemos:

<table>
<thead>
<tr>
<th>Riesgo</th>
<th>Criticidad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baja de algún miembro del equipo</td>
<td>Alta</td>
</tr>
<tr>
<td>Cese de colaboración de nuestro contacto en IBM</td>
<td>Media</td>
</tr>
<tr>
<td>Desfase temporal con impacto sobre el resto del proyecto</td>
<td>Media</td>
</tr>
<tr>
<td>Pérdida parcial/total de los elementos desarrollados por catástrofe</td>
<td>Media-Alta</td>
</tr>
</tbody>
</table>

Cuadro 10.1.2: Los posibles riesgos que acechan al proyecto

Para el caso de la materialización del primero de los riesgos, el impacto sobre el proyecto hubiera sido mínimo puesto que cada miembro desarrolla partes muy diferenciadas. Por la manera de trabajar, los dos miembros que conforman este equipo definen en conjunto el modelo del nivel PSM para una tecnología concreta, estudiando previamente la tecnología sobre la que se va a implementar. En el caso de cumplirse este riesgo, se hubiera dificultado la definición de dicho modelo pero no hubiera impactado sobre el resto de fases. Además, parte de este documento es compartida entre la memoria de ambos miembros, por lo que el tiempo que hubiera sido necesario para obtenerlo hubiera sido mayor.

En el caso de llevarse a cabo el segundo de los riesgos que exponemos, hubiera tenido gran repercusión al inicio del proyecto puesto que la documentación pública que hay al respecto de cómo elaborar transformaciones en Rational Software Architect utilizando la librería UML2 es mínima. Aún así a medida que el tiempo avanza este riesgo disminuye.

Para minimizar al máximo el impacto sobre el proyecto del tercero de los riesgos, decidimos que la mejor manera de trabajar consiste en llevar a cabo un proceso iterativo trabajando de forma bottom-up. De este modo, se minimiza el tiempo necesario para cada fase tal y como explicamos en el apartado correspondiente. Al trabajar de forma iterativa, se pueden detectar rápidamente posibles desviaciones temporales y se pueden tomar medidas al respecto.

Para impedir la materialización del último riesgo expuesto, acondicionamos un servidor CVS en una tercera máquina. A parte de tener copias locales de todos los ficheros que conforman este trabajo en las máquinas de los dos miembros del equipo, el hecho de tener un CVS nos permite poder recuperar versiones anteriores de cualquier documento.
10.2. PLANIFICACIÓN *

Periódicamente, se ha llevado a cabo copias de seguridad de los ficheros, por lo que podemos decir que esta situación no puede afectar al proyecto.

10.2. Planificación *

En esta sección detallaremos el método de planificación que se ha seguido. En los breves inicios del proyecto, se plantearon diversas posibilidades. De entre ellas contemplamos:

- Diagrama de Gantt
- Diagrama de Pert
- Diagrama de Roy

**Decisión:**
Valorando la diversas alternativas, decidimos establecer la planificación en un diagrama de Gantt puesto que es sencillo de realizar, sencillo de comprender, y propone de forma visual muy intuitiva el orden de precedencia de tareas.

10.2.1. Diagrama de Gantt

Haremos un listado de las tareas (con sus respectivas subtareas) descritas en el diagrama de Gantt realizado para el proyecto.

- Reuniones
  - Reuniones de seguimiento con Álex y Ernest
  - Reuniones de seguimiento con Ferran

- Documentación
  - Introducción
  - Objetivos
  - MDA
  - IBM Rational Software Architect
  - UML2
  - Desarrollo de una transformación
  - Plugins desarrollados
  - Manuales de uso de los plugins
  - Metodología y Planificación
• Conclusiones
• Otros

■ Instalación de software y mantenimiento
  • IBM Rational Software Architect

■ Estudio
  • Patrones de diseño
  • MDA
  • Rational Software Architect
  • Otros

■ Modelos
  • CRUD
    ○ Struts
    ○ PHP - Descartado
  • Login
    ○ Java Swing

En la figura 10.2.1 podemos ver una representación gráfica del Gantt en cuestión.

10.2.2. Planificación temporal *

El proyecto se inició oficialmente la semana del 3 de Julio de 2006. Considerando que debía estar presentado como fecha límite el 2 de Febrero de 2007, consideramos la planificación que se muestra en el siguiente cuadro.

Treinta semanas de duración del proyecto, con una dedicación de 25 horas semanales, nos permiten aproximarnos a las 750 horas que debe suponer un proyecto final de carrera de 37.5 créditos.

Como se muestra en la tabla, en algunos apartados la planificación temporal ha sido bastante acertada, mientras que en otros la desviación ha sido considerable. En el caso de la documentación, ha resultado bastante más costoso de lo que se planeó inicialmente.

Puesto que el proyecto se complementa con el de Ferran Basora Roca, decidimos hacer reuniones semanales de 3 horas para la toma de decisiones, compenetración del equipo y corregir desviaciones. Esta fase también ha durado más de lo previsto, puesto que a menudo bastantes de las decisiones que se han tomado han influido activamente en ambos proyectos.
## 10.2. PLANIFICACIÓN *

<table>
<thead>
<tr>
<th>Grupo</th>
<th>Tarea</th>
<th>Horas estimadas</th>
<th>Horas reales</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preparar entorno</td>
<td>-</td>
<td>25h</td>
<td>45h</td>
</tr>
<tr>
<td>Estudio</td>
<td>Rational Software Architect</td>
<td>40h</td>
<td>65h</td>
</tr>
<tr>
<td></td>
<td>Model Driven Architecture</td>
<td>60h</td>
<td>40h</td>
</tr>
<tr>
<td>Diseño del proceso</td>
<td>CRUD</td>
<td>60h</td>
<td>80h</td>
</tr>
<tr>
<td></td>
<td>Login</td>
<td>50h</td>
<td>40h</td>
</tr>
<tr>
<td>Implementación</td>
<td>CRUD Struts</td>
<td>90h</td>
<td>100h</td>
</tr>
<tr>
<td></td>
<td>CRUD PHP</td>
<td>20h</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Login</td>
<td>80h</td>
<td>50h</td>
</tr>
<tr>
<td></td>
<td>Model Utility</td>
<td>0h</td>
<td>30h</td>
</tr>
<tr>
<td>Documentación</td>
<td>Memoria</td>
<td>210h</td>
<td>230h</td>
</tr>
<tr>
<td></td>
<td>Presentación</td>
<td>25h</td>
<td>15h</td>
</tr>
<tr>
<td>Reuniones</td>
<td>Reuniones de seguimiento</td>
<td>90h</td>
<td>115h</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td>750h</td>
<td>810h</td>
</tr>
</tbody>
</table>

**Cuadro 10.2.1:** La planificación temporal de este proyecto
Capítulo 11

Conclusiones

A lo largo de este capítulo veremos las conclusiones que se extraen como resultado de la elaboración del proyecto. Veremos también los beneficios empresariales que puede aportar la implantación de MDA, y el trabajo futuro que se plantea.

11.1. Conclusiones del proyecto *

A lo largo de todo el proyecto se han visto los enormes beneficios de desarrollar aplicaciones basándose en el Model-Driven Development. Por una parte se ha visto que resulta un proceso mucho más sencillo de cara al usuario. Análogamente, sería una solución parecida a lo que aportan los patrones de diseño. ¿Cuál es el sentido de analizar y diseñar algo que puede ser reutilizado?

Aún así, no todo es tan sencillo como puede parecer. Los modelos diseñados para ser utilizados en un proceso MDD deben ser lo suficientemente flexibles para que puedan ser reutilizados por otras personas. Si las transformaciones diseñadas generan una solución demasiada concreta, serán útiles para un rango más reducido de usuarios.

El paradigma MDA es un estándar demasiado abierto. Ello nos ha llevado en numerosas ocasiones a dudar de cómo plantear las soluciones descritas. Hemos encontrado diversas interpretaciones que se ajustan a la definición de dicho estándar.

El desarrollo de plugins en Rational Software Architect puede ser un proceso complejo. El hecho de utilizar la librería UML2 puede suponer una barrera puesto que hay poca información de cómo utilizarla correctamente. Hay a disposición de cualquier usuario una descripción de clases y funciones generada con Javadoc, pero puede resultar poco intuitiva para alguien con poco conocimiento del tema.

RSA es un entorno muy pesado. Realizar las transformaciones con sus correspondientes pruebas puede resultar un proceso extremadamente lento en máquinas no demasiado
potentes. A medida que trabajamos con modelos más grandes, el rendimiento del sistema cae en picado. Es un punto de mejora del software.

Aún así, hemos visto que el potencial y la flexibilidad de RSA es muy elevado. Permite diseñar cualquier tipo de aplicación facilitando enormemente el trabajo al usuario. Los tutoriales que integra la herramienta son sumamente descriptivos y fáciles de seguir.

Respecto la gestión de proyectos, hemos contemplado que la planificación realizada a priori nunca se cumple. Se van materializando riesgos y en ocasiones el tiempo estimado de duración de una tarea es mayor (y en otras menor) al previsto. Es por ello que ha resultado muy beneficioso el desarrollo iterativo. Gracias a las reuniones periódicas que hemos realizado, hemos podido reconducir la planificación y tomar decisiones sobre la marcha para garantizar un resultado exitoso.

11.2. Beneficios empresariales de implantar MDA

En este apartado responderemos a la siguiente pregunta: ¿merece la pena que las empresas desarrollen basándose en nuestro producto?

La inversión inicial que debería hacer una empresa para aprovecharse de MDA es:

<table>
<thead>
<tr>
<th>Producto</th>
<th>Valor</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM Rational Software Architect Licencia Flotante + 12 meses mantenimiento</td>
<td>11.000 €</td>
</tr>
<tr>
<td>Formación de MDA de 4 analistas (2 semanas)</td>
<td>3.000 €</td>
</tr>
<tr>
<td>Formación de MDA de 4 diseñadores (2 semanas)</td>
<td>3.600 €</td>
</tr>
<tr>
<td><strong>Total:</strong></td>
<td><strong>17.600 €</strong></td>
</tr>
</tbody>
</table>

Cuadro 11.2.1: Inversión que debería hacer una empresa para implantar MDA

Hemos supuesto un coste diario de 75 € diarios por cada analista, y 90 € por cada diseñador.

La licencia que contemplamos permite que varios usuarios de un mismo equipo de desarrollo puedan trabajar con la misma licencia (pero sólo uno de forma simultánea).

Si repartimos los costes entre cada uno de ellos, tenemos que cada uno de los trabajadores formados cuesta 2.200 €. Para amortizar los gastos invertidos en cada analista, las transformaciones MDA deben ahorrar 30 días de trabajo. En el caso de los diseñadores, a partir de 24 días de trabajo ahorrado obtendríamos beneficios. No debemos olvidar que nuestras transformaciones generan código que funciona correctamente a partir del diseño resultante. Es por ello que un modelo MDD ahorra muchas horas de programación.

Si suponemos que un programador cuesta 60 € diarios y nuestra aplicación requiere de un CRUD con Struts, suponiendo que sea necesario que 2 programadores trabajen durante 1 semanas para crear el CRUD de una entidad tendría un coste de 600 €. Deduciéndolo
11.3. TRABAJO FUTURO *

del coste total anteriormente mostrado, tenemos que se necesitan 27 días ahorrados por
analista y 22 días por cada diseñador para amortizar la inversión.
Basándonos en el resultado anterior, con sólo aplicar las transformaciones diseñadas pa-
ra mantener 30 entidades utilizando MDA para el CRUD en Struts, sólo en horas de
programación amortizamos el coste de inversión de RSA y aprendizaje de analistas y
diseñadores.
Si aplicamos otros modelos como el login y otros que se puedan desarrollar, la amortiza-
ción se producirá mucho antes.

11.3. Trabajo futuro *

- Las implementaciones llevadas a cabo en este proyecto no están lo suficientemente
  perfeccionadas como para funcionar correctamente en todos los casos. El proceso
descrito explica qué tiene que hacer un usuario que quiera llevar a cabo un desarro-
lllo MDD, pero la transformación no hace todas las comprobaciones para garantizar
que se están llevando a cabo las restricciones descritas en el proceso. Como trabajo
futuro propongo perfeccionar dichas transformaciones para impedir que el usuario
pueda provocar una ejecución anómala de la transformación. Esto es así porque el
objetivo de este proyecto es estudiar el método de adaptación de MDA en RSA, por
lo que la robustez del producto desarrollado juega un papel secundario.

- Otro de los aspectos a mejorar es que las transformaciones desarrolladas no están
  a priori preparadas para trabajar conjuntamente entre ellas. Las pruebas llevadas a
cabo han sido aplicando las transformaciones de modo unitario. Es por ello que no
hay garantías de que, al trabajar conjuntamente, el comportamiento sea adecuado.

- Se podría mejorar la interacción máquina-persona, desarrollando para cada trans-
  formación un proceso asistido que orientara al usuario lo que debe hacer para poder
conseguir una transformación. De la manera que se ha llevado a cabo el proceso
da una alta libertad al usuario, y debe ceñirse al método descrito para garantizar
resultados correctos.

- Hay ideas de modelos MDA que han ido surgiendo a lo largo de la elaboración del
  proyecto. Una de ellas fue realizar la modelización de una aplicación J2ME1 me-
diante el proceso asistido MDD. Gracias al desarrollo MDD, podríamos tener una
aplicación modelada en el nivel PIM que pudiera servir tanto para una aplicación
J2ME como para otra J2EE. Sería responsabilidad de la transformación del PIM al
PSM transformar el modelo a las particularidades de cada uno de ellos.

- Se propone otra vía para avanzar el estudio que se ha realizado en este trabajo. Ya
  está disponible la versión 7.0 de Rational Software Architect. El método descrito

---

1 La rama del entorno de desarrollo Java para dispositivos móviles
en esta memoria no está probado en dicha versión del software. Se propone como trabajo futuro adaptar el proceso para que funcione en la versión 7.0.
Índice de figuras

2.2.1. Ciclos de RUP .................................................. 7
2.3.1. Ciclo de desarrollo ........................................... 7
2.3.2. Modelos Arquitectónicos .................................. 9
2.4.1. Mapeo niveles MDA y artefactos RUP .................. 10

4.1.1. Modelo MDA ..................................................... 16
4.3.1. Niveles MDA ................................................... 20
4.4.1. Transformación de modelos ............................... 21
4.4.2. Transformación PIM-PSM ................................... 22
4.4.3. Transformación de un modelo del paradigma MVC .... 22
4.5.1. Metamodelo Base de datos relacional .................... 24
4.5.2. Metamodelo de clases ....................................... 25
4.5.3. Metamodelo Base de datos relacional .................... 25

5.3.1. Entorno de trabajo en RSA ................................. 31
5.3.2. Model Explorer ............................................... 32
5.5.1. Contenido de un profile .................................... 34
5.5.2. Listado del patrones en RSA ............................... 35
5.5.3. Configuraciones de lanzamiento ......................... 38

6.1.1. Diagrama de secuencia de una transformación .......... 43

7.2.1. Creación de un profile (1) ................................. 57
Índice de cuadros

2.2.1. Artefactos básicos en RUP ................................................. 6
2.2.2. Roles ................................................................. 7
8.3.1. Atributos parametrizables del PSM Swing para Login .......... 77
10.1.1. Detalles de las fases de desarrollo ............................. 107
10.1.2. Lista de riesgos .................................................... 108
10.2.1. Planificación temporal .......................................... 111
11.2.1. Coste de una empresa que implante MDA .................... 114
Glosario

ATL  ATLAS Transformation Language
DAO  Data Access Object
DSL  Domain Specific Languages
EDOC  Enterprise Distributed Object Computing
ERP  Enterprise Resource Planning
GUI  Graphics User Interface
IDE  Integrated Development Environment
MDA  Model Driven Architecture
MDD  Model-Driven Development
MDE  Model-Driven Engineering
MOF  Meta-Object Facility
OCL  Object Constraint Language
OMG  Object Management Group
Patrón  Abstracción de una manera de hacer o resolver un problema
PIM  Platform Independent Model
PDM  Platform Definition Model
PSM  Platform Specific Model
RUP  Rational Unified Process
QVT Queries/Views/Transformations
XMI XML Metadata Interchange
UML Unified Modelling Language
UP Unified Process
Bibliografía


[Patrón de diseño] Disponible en: http://es.wikipedia.org/wiki/Patr%C3%B3n_dise%C3%B1o


[OpenMDX] *OpenMDX*. Disponible en:
http://www.openmdx.org/

[Object-Oriented Systems Analysis: Modelling the World in Data] Sally Shlaer, Stephen Mellor. [Varias consultas].

[Model Driven Architecture (MDA)] Miller, J., Mukerji, J., others. [Varias consultas].

