PEDECIBA Informática
Instituto de Computación – Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay

Reporte Técnico RT 06-15

Modelo de desarrollo de software OO. Experimentación en un curso de Ingeniería de Software

Andrea Delgado Beatriz Pérez

2006

Modelo de desarrollo de software OO. Experimentos en un curso de Ingeniería de Software.
Delgado, Andrea; Pérez Beatriz
ISSN 0797-6410
Reporte Técnico RT 06-15
PEDECIBA
Instituto de Computación – Facultad de Ingeniería
Universidad de la República

Montevideo, Uruguay, 2006

Modelo de Desarrollo de Software OO
Experimentación en un curso de Ingeniería de Software [ϕ]

Andrea Delgado, Beatriz Pérez

Universidad de la República,
Instituto de Computación,
Grupo de Ingeniería de Software,
Montevideo, Uruguay
{adelgado,bperez}@fing.edu.uy

Febrero 2006

Resumen

En el grupo de Ingeniería de Software (Gris) del Instituto de Computación de la Facultad de Ingeniería de la Universidad de la República – Uruguay, el eje central de las actividades está constituído por el Programa de construcción y prueba de modelos de proceso.

En este artículo se presenta la concepción del Modelo de Desarrollo de Software Orientado a Objetos y cómo este proceso ha evolucionado en los últimos cinco años. Se presentan sus principales características, los productos que se han construido siguiendo el modelo, las conclusiones de su aplicación y los motivos de sus ajustes y mejoras.

Palabras clave: Ingeniería de Software, procesos y metodologías de desarrollo de software, experimentación en Ingeniería de Software, aplicación y mejora de procesos.

[ϕ] Artículo publicado en las "V Jornadas Iberoamericanas de Ingeniería de Software e Ingeniería del Conocimiento (JIISIC'06)", Puebla, México, Febrero de 2006. ISBN 970-94770-0-5

1 Introducción

La construcción de un producto de software involucra varias etapas y actividades y el orden en que éstas se realizan definen el ciclo de vida del software. El proceso seguido para desarrollar, liberar y evolucionar en distintas versiones un producto de software, desde la concepción de una idea hasta la concreción de la misma se conoce como el proceso de software. En general, un proceso disciplinado es seguido en forma consistente cuando todos los participantes entienden el valor de hacerlo y existe en la Organización la infraestructura necesaria para brindar el soporte requerido [1].

El programa de construcción y prueba de modelos de proceso constituye el eje central de las actividades del grupo de Ingeniería de Software (Gris) de la Facultad de Ingeniería de la Universidad de la República del Uruguay[2]. El programa se inició en el año 2000 y se puede consultar en [3].
En este artículo se presenta la concepción del Modelo de Desarrollo de Software Orientado a Objetos y cómo este proceso ha evolucionado en los últimos cinco años. Se presentan sus principales características, los productos que se han construido siguiendo el modelo, las conclusiones de su aplicación y los motivos de sus ajustes y mejoras.

En la sección 2 se describe la experimentación que se realiza en un curso de Ingeniería de Software y la evolución del Modelo de Desarrollo OO, en la sección 3 se presenta este Modelo, en la sección 4 se describe el seguimiento del programa de prueba de procesos y la estructura organizacional necesaria para mantenerlo, en la sección 5 se presentan una relación de los ajustes realizados al proceso en cada año, a partir del 2000 y en la sección 6 se presentan las conclusiones.

2 Experimentación en Ingeniería de Software

El programa construcción y prueba de modelos de procesos es dirigido por docentes del Grupo de Ingeniería de Software y para implementarlo se trabaja con estudiantes de la carrera de Ingeniero en Computación. El programa cuenta con tres hitos principales: obtener un proceso definido, obtener un proceso validado y obtener un proceso ajustado y mejorado. La definición de los procesos se realiza por estudiantes del quinto año de la carrera que cursan la asignatura “Proyecto de Grado” durante el primer semestre de clases, obteniendo como resultado el proceso definido, su puesta en práctica se hace durante el segundo semestre, con estudiantes de cuarto año en la asignatura “Proyecto de Ingeniería de Software”[4]. Los estudiantes de “Proyecto de Grado” definen el proceso, y durante el segundo semestre asisten en su comprensión y utilización por parte de los estudiantes, a la vez que evalúan el apego al proceso por parte de los grupos, detectando problemas en el mismo mediante la realización de Auditorias al proceso.

La prueba de estos procesos se realiza en la asignatura “Proyecto de Ingeniería de Software” la cual tiene criterios establecidos que se describen a continuación: la duración del proyecto es de 14 semanas, los grupos de estudiantes son de entre 8 y 13 personas y la carga horaria prevista por estudiante es de 15 horas semanales. Los docentes del curso son los encargados de dirigir a los grupos cumpliendo el rol de “director”, los grupos tienen una agenda definida donde se pautan para cada semana, las principales actividades a realizar y los productos que se espera se entreguen al cliente y al director. El producto se instala en el ambiente del cliente en las últimas dos semanas del proyecto y no se realiza mantenimiento del mismo.

Al terminar el segundo semestre, se alcanza el segundo hito: obtener un proceso validado. Luego se evalúa su aplicación en los proyectos y los resultados obtenidos, identificando mejoras al proceso definido, obteniendo un proceso ajustado y mejorado, el cual puede ser puesto en práctica al siguiente año.

2.1 Concepción y evolución de procesos

En la figura 1 se muestra la evolución de los procesos, a partir del año 2000 cuando comenzó el programa. Los procesos que se describen en esta sección pueden ser consultados en [4]. En el año 2000 se definieron y probaron dos modelos de proceso por parte de dos grupos de estudiantes de "Proyecto de Grado", nombrados Modelo de Proceso Java I (MP Java I) y Modelo de Proceso Java II (MP Java II), que se basaron en el Proceso Unificado (UP) de Booch, Jacobson y Rumbaugh [5]. Sin embargo, en el UP solamente se describen las actividades, entregables y roles para las tareas "tradicionales" de desarrollo de software dadas por: Requerimientos, Análisis, Diseño, Implementación y Verificación, por lo que para la definición de las disciplinas de Gestión de Calidad, Gestión de Configuración y Gestión del Proyecto, se utilizaron también otras fuentes como los Modelos de Mejora de Procesos Capability Maturity Model (CMM) [1], y Software Process Improvement and Capability Determination (SPICE) [6], la metodología definida para la administración pública española Métrica V.3 [7] y el Adaptable Process Model (APM) de Roger Pressman [8].

En el año 2001, se fusionaron ambos modelos en el MP Java. Dicho modelo tuvo como aportes además de los anteriores, el Rational Unified Process (RUP) [9], que evoluciona el Unified Process incluyendo actividades de gestión. Hasta ese momento, los clientes eran docentes del curso, en el 2001 se empezó a trabajar con clientes reales.

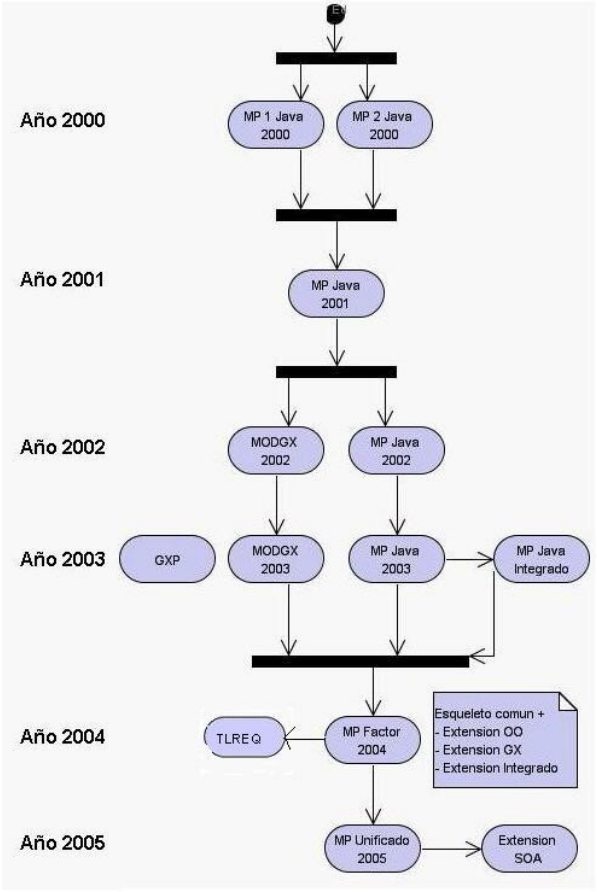

Figura 1 – Evolución de los procesos en el tiempo

A partir de la evaluación de la experiencia del 2001 en que se introdujeron los clientes reales, se decidió incorporar en el 2002 a los procesos utilizados, la fase de transición, con el objetivo de entregar el producto al usuario, por más que a estos no se les aseguraba que el producto final pudiera ponerse en explotación sin esfuerzo adicional [3]. Se definió además MoDSGX para desarrollo de software con la herramienta GeneXus [10] basado en el MP Java 2001 y en RUP.

En el año 2003 se puso en práctica el MP Java que ajustaba y mejoraba la versión probada en el 2002, y el MoDSGX en un segundo ciclo de desarrollo. Además se construyó una extensión al MP Java denominado Integrado, para permitir el trabajo conjunto de más de un grupo, desarrollando en común un producto único y se realizó una adaptación de Extreme Programming (XP) [11], metodología de desarrollo ágil propuesta por Kent Beck, la adaptación fue llamada GXP y adaptaba XP a las particularidades del curso y a la herramienta de desarrollo Genexus.

En ese mismo año se detectó un problema con el programa de prueba de procesos, dado que MoDSGX se basa en el MP Java 2001, y que ambos procesos evolucionaron, cada cambio que se decidía realizar, debía impactarse en ambos procesos, los cuales eran mantenidos por personas distintas. Debido a esto en el 2004 se definió un proceso que fusionó los procesos MoDSGX y MP Java en el proceso Factor, el cual tiene un esqueleto común a ambos procesos de actividades, entregables y roles, e instancias particulares para desarrollo con Genexus (extensión GX) o para desarrollo orientado a objetos (extensión OO). En la edición 2004 también se puso a prueba una extensión de Factor para desarrollos OO llamada TLReq, para el caso en el cual el cliente y el equipo de desarrollo se encuentran alejados, por ejemplo en distintos países.

En la edición 2005, se quiere mejorar la calidad del proceso Factor, principalmente en las disciplinas de calidad y verificación, incluyendo el estudio de la satisfacción del cliente como parte del proceso. Esta edición del proceso se llama Unificado. También se ha realizado este año un agregado al proceso Unificado en su extensión OO, para desarrollos con enfoque SOA (Service Oriented Architecture) que plantea una metodología para desarrollo de aplicaciones orientadas a servicios.

3 Modelo de Desarrollo de Software OO

El Modelo de Desarrollo de Software OO es como se menciono en la sección 2, principalmente una adaptación del Unified Process (UP), actualmente conocido como Rational Unified Process (RUP). Este proceso plantea un enfoque de desarrollo iterativo incremental que permite un entendimiento progresivo de los requerimientos del Sistema a través de sucesivos refinamientos y el crecimiento incremental de una solución efectiva al problema planteado, permitiendo también que los riesgos del proyecto sean identificados en cada etapa del desarrollo, ayudando a reducirlos significativamente. El desarrollo está basado en Casos de Uso para capturar los requerimientos y asegurar que éstos guían el diseño, implementación y verificación del software. Es centrado en la Arquitectura del Software, realizando la definición y construcción de ésta en etapas tempranas del desarrollo, lo que permite reducir los riesgos tecnológicos asociados, y sobre la cual se construirán incrementalmente el resto de las funcionalidades del sistema. Para el modelado en las disciplinas se utiliza el lenguaje UML.

La estructura del Modelo Java se describe en dos dimensiones, siguiendo la definición de UP: la dimensión del tiempo que muestra los aspectos dinámicos del proceso y se representa mediante ciclos, fases, iteraciones y objetivos, y la dimensión del modelo que muestra los aspectos estáticos del proceso y define Disciplinas con sus actividades y entregables (artefactos), y roles.

3.1 Dimensión del tiempo

En el Modelo que propone UP, el desarrollo del software se divide en ciclos donde cada ciclo trabaja para construir una nueva Generación del Sistema y a su vez cada ciclo se divide en cuatro fases: Inicial, Elaboración, Construcción y Transición. A los efectos de su aplicación en el año 2000, donde el Sistema construido no llegaba a ser verificado por el Cliente ni puesto en producción, el modelo propuesto originalmente fue adaptado con un solo ciclo donde se cumplen las siguientes fases: Inicial, Elaboración y Construcción. A partir del año 2001 se introdujeron clientes reales y se vio que era necesario realizar actividades de implantación al cliente, por lo que a partir del curso del año 2002 se agregó al modelo la fase de Transición con actividades que contemplan la transición al ambiente del cliente como instalación y pruebas de aceptación. Cada fase se divide en iteraciones y tiene objetivos definidos que al cumplirlos indican su finalización. La forma de visualizar que se alcanzaron esos objetivos es por intermedio de los entregables de la fase.

3.2 Agenda

Para adaptar el modelo al curso se define también una Agenda en la cual se indica la duración de cada fase e iteración en semanas, de acuerdo a la duración total del curso, indicando para cada semana las actividades del proceso que se deben realizar y los entregables a generar. La duración del proyecto es una variable fija que no se puede modificar, siendo en los años 2000 y 2001 de 13 semanas, y a partir del año 2002 de 14 semanas. La duración de las Fases e iteraciones fue evolucionando desde la primer Agenda realizada para la edición del año 2000, variando principalmente la cantidad de iteraciones y duración de las mismas en las Fases Inicial y de Construcción, y el agregado de la Fase de Transición para la edición 2002. La Agenda definida para ese año, se mantiene hasta el momento como se muestra en la figura 2.

<table>
<tr><td>Semana /Año</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td></tr>
<tr><td rowspan="2">2000</td><td colspan="2">Inicial</td><td colspan="4">Elaboración</td><td colspan="6">Construcción</td><td>X</td></tr>
<tr><td colspan="2">Iter. 1</td><td colspan="2">Iter. 1</td><td colspan="2">Iter. 2</td><td colspan="3">Iter. 1</td><td colspan="3">Iter. 2</td><td>X</td></tr>
<tr><td rowspan="2">2001</td><td colspan="3">Inicial</td><td colspan="3">Elaboración</td><td colspan="5">Construcción</td><td>X</td></tr>
<tr><td colspan="2">Iter. 1</td><td colspan="2">Iter. 2</td><td colspan="2">Iter. 1</td><td colspan="2">Iter. 2</td><td colspan="3">Iter. 1</td><td colspan="2">Iter. 2</td><td>X</td></tr>
<tr><td rowspan="2">2002 a 2005</td><td colspan="3">Inicial</td><td colspan="3">Elaboración</td><td colspan="4">Construcción</td><td colspan="3">Transición</td></tr>
<tr><td colspan="2">Iter. 1</td><td colspan="2">Iter. 2</td><td colspan="2">Iter. 1</td><td colspan="2">Iter. 2</td><td colspan="2">Iter. 1</td><td colspan="2">Iter. 2</td><td colspan="2">Iter. 1</td></tr>
</table>

Figura 2: Evolución de la duración de las Fases

3.3 Dimensión del modelo

La dimensión del modelo se basa en cuatro elementos de modelado principales para describir en el proceso definido, quién está haciendo qué, de que forma (como) y en que momento (cuando). Estos elementos son las Disciplinas que describen cuando, los roles que describen quién, las actividades que describen como y los entregables que describen qué. Cada Disciplina representa una división de Actividades, las cuales son agrupadas en forma lógica. Para cada Disciplina se describe la secuencia de actividades y los roles que interactúan para obtener los distintos Entregables. Las Disciplinas Básicas son las que involucran las actividades de ingeniería "tradicionales" de desarrollo de software las cuales se realizan como mini-cascadas en cada iteración y están divididas en: Requerimientos, Análisis, Diseño, Implementación y Verificación. Las Disciplinas de Gestión están constituidas por actividades que brindan "soporte" a las Disciplinas Básicas y se realizan en forma paralela a éstas en cada iteración, dividiéndose en: Gestión de Configuración (SCM), Gestión de Calidad (SQA) y Gestión del Proyecto (GP).

3.4 Roles y combinación de roles

Un rol define el comportamiento y las responsabilidades de una persona ó un grupo de Personas trabajando en equipo. Una persona puede tener varios roles y un mismo rol puede ser cumplido por más de una persona. El comportamiento está dado por las actividades en las que participa el Rol, y las responsabilidades están dadas por el conjunto de Entregables que el Rol tiene asociado. Se incluyen en el Modelo OO los siguientes roles: Analista, Arquitecto, Especialista Técnico, Implementador, Responsable de Verificación, Administrador, Responsable de SQA, Responsable de SCM, Documentador de Usuario y Asistente de Verificación.

A partir de estos roles se definen los roles combinados para el proceso, teniendo en cuenta aspectos como carga de trabajo de los roles en el tiempo, compatibilidad de tareas o enfoque de las áreas en las que participan, de forma que cada persona en el proyecto cumple un rol combinado que determina su participación en el mismo.

3.5 Actividades

Una actividad es un conjunto de acciones que se realizan en una Disciplina, con el objetivo de crear o actualizar uno o varios entregables pertenecientes a la misma. Cada actividad se compone de un conjunto de entregables de entrada que son las pre-condiciones para su ejecución, un conjunto de roles que la realizan que son los participantes de la misma, una descripción de las tareas a realizar en la actividad y un conjunto de entregables de salida que son las post-condiciones de su ejecución. En cada Disciplina las actividades tienen un flujo de ejecución definido para su realización en cada iteración y Fase, algunas pueden hacerse en paralelo, y también entre Disciplinas distintas. Según el momento del proyecto en que se esté se podrán realizar todas, algunas o ninguna de las actividades definidas siguiendo el flujo establecido.

3.6 Entregables

Los entregables son los productos tangibles del Proyecto, los cuales son entrada y salida de las distintas actividades. Cada uno tiene un contenido definido, propiedades de calidad que debe cumplir y, a partir del año 2001 se incorporó una plantilla como guía para realizarlo. En la plantilla se incluye información para cada sección definida para el documento, donde se describe cual es el contenido esperado en cada caso. Tiene asociado un rol responsable de su existencia, contenido y mantenimiento.

4 Seguimiento del Modelo

Para el seguimiento y evaluación de cada Modelo se realizan las siguientes actividades por parte de los docentes del Grupo de Ingeniería de Software (GrIS).

4.1 Revisión con el Director

Semanalmente se realiza una Revisión con el Director del Proyecto, que como ya se mencionó es un docente asignado al curso, a la que deben asistir en forma obligatoria solo los roles de Administrador, Arquitecto, y Responsables de Calidad y de Verificación, elegidos porque representan la visión de las distintas áreas de trabajo en el proyecto, para poder ver el avance del proyecto con toda la información del mismo y de forma que sea aprovechable por el grupo sin incurrir en gasto de horas de todos los integrantes. El resto de los roles pueden asistir a cualquier Revisión si tienen alguna duda, y todos los integrantes del grupo deben asistir a las revisiones correspondientes a fin de Fase, para discutir con el director la evaluación realizada y el pasaje o no a la Fase siguiente según el cumplimiento de objetivos logrado. El Director de Proyecto realiza el monitoreo de los Entregables más importantes según la Fase e iteración. El curso tiene además, una reunión de coordinación semanal entre todos los Directores de Proyecto y el responsable del curso, en la cual cada Director presenta un resumen de la revisión con cada uno de sus grupos y de su visión del estado del proyecto en cada caso, poniendo en común problemas para resolver entre todos los docentes y experiencias a tener en cuenta. Esto permite además nivelar el trabajo de los distintos grupos.

4.2 Auditorias del proceso

La IEEE define una Auditoria como "la revisión independiente de un producto o de un conjunto de productos para evaluar el cumplimiento con las especificaciones, estándares, acuerdos contractuales u otros criterios" [12]. Las auditorias son realizadas por docentes del curso y por los estudiantes que definieron el proceso. Para la realización de las Auditorias, se identifican las Disciplinas que a ser auditadas en cada Fase y/o iteración, teniendo en cuenta las actividades claves establecidas para cada Disciplina en cada momento del proceso y la Agenda de Actividades y Entregables definida en el modelo. La presencia de los Entregables de salida de una actividad indica en principio, la realización de la misma.

De todas formas el hecho de que exista el Entregable no garantiza que la actividad se haya realizado siguiendo la descripción dada, aunque su contenido da una idea de las tareas realizadas para su generación. Luego de definidas las Disciplinas que serán auditadas, se identifican los roles involucrados y se convoca a la Auditoria mediante un documento en el cual se detalla la temática y las preguntas a realizar. Posteriormente a las Auditorias se realiza un informe de la reunión el cual se envía a los participantes para que confirmen su acuerdo con la información incluida, luego se envía también a los Directores de Proyecto para que tengan conocimiento de la visión externa de sus grupos. En los distintos años se ha constatado que las Auditorias sirven a los grupos como clases de consulta donde se les explica la razón de hacer determinadas actividades y entregables.

4.3 Encuestas de satisfacción

Al finalizar el proyecto se realiza el cierre por parte de los grupos con informes finales en las áreas de Verificación, Gestión del Proyecto, de la Calidad, y Configuración, en los que se incluyen datos interesantes del desarrollo del mismo. También se envían cuestionarios finales a estudiantes, directores y clientes, en los que se realizan preguntas sobre el modelo de proceso y desarrollo del proyecto por parte del grupo, y cuyas respuestas son procesadas en forma manual y utilizadas para ajustar el modelo para la siguiente edición del curso [13].

Actualmente se está en vías de automatizar el procesamiento de los cuestionarios para que puedan ser ingresados directamente por parte de los estudiantes, director y cliente desde la Web, registrando y almacenando las respuestas en el momento en una base de datos que permita extraer fácilmente los datos relevantes para realizar el ajuste al Modelo.

4.4 Memoria Organizacional

Los sistemas de Memoria Organizacional están conectados al concepto de la Gestión del Conocimiento ya que es el tipo de sistema utilizado para llegar al nivel de gestionar el conocimiento en la organización, también está conectado al concepto de Aprendizaje Organizacional pero la distinción con el Aprendizaje Organizacional es que en éste el foco está dado en el aprendizaje a partir de experiencias previas.

Aprender de la experiencia en proyectos pasados es una forma de mejorar la calidad del software en nuevos proyectos. Esto puede implicar la reutilización de técnicas, métodos, herramientas, procesos, o métricas usadas en otros proyectos. El aprendizaje no se limita sólo a proyectos exitosos, las fallas especialmente las ocurridas en proyectos pasados deben ser cuidadosamente analizadas y documentadas para poder evitarlas en el futuro. Para poder reutilizar las experiencias pasadas éstas deben ser recolectadas y almacenadas.

El enfoque de aprendizaje organizacional basado en el desarrollo de software recolecta información relacionada con el proyecto durante la creación de productos individuales de software. Esta información puede ser luego diseminada hacia proyectos subsiguientes para proveer conocimiento basado en la experiencia relativo a temas del desarrollo encontrados en la Organización. Este conocimiento está dado por "lecciones aprendidas", tips y técnicas para desarrollar tareas específicas, código reusable, entre otros. [13]

La Memoria Organizacional es un sitio web, donde en cada año se almacenan los procesos que se pusieron en práctica y los entregables generados por los grupos, de forma que estén disponibles para siguientes ediciones como documentación de referencia y consulta.

5 Ajustes al Modelo

En base a las pruebas del modelo realizadas en las distintas ediciones del curso, se van realizando ajustes al modelo y se aprende de las distintas experiencias. Para realizar estos ajustes se toman en cuenta los resultados de las Auditorias realizadas a los grupos, así como los cuestionarios que se realizan al finalizar el curso, a los estudiantes y directores de proyecto para evaluar el modelo de proceso seguido y el curso, y a los clientes para evaluar los productos obtenidos y aspectos del curso relevantes al rol.

A continuación se describen los principales ajustes realizados según el año al que corresponde la experiencia de aplicación del modelo. En las tablas que se presentan para cada experiencia se muestran los datos de la experiencia indicando los grupos que cursaron la asignatura, cantidad de integrantes, el total de líneas de código LOCs generadas y las horas de esfuerzo incurridas por cada uno, así como el modelo de desarrollo que siguió el grupo y quién cumplió el rol de cliente con cada uno.

5.1 Experiencia 2000

En la experiencia 2000 todos los grupos desarrollaron el mismo proyecto con un único cliente. Desde el punto de vista del cliente, el hecho de tener más de un grupo desarrollando el mismo proyecto, si bien implica un esfuerzo adicional, también ofrece mayores chances de obtener un resultado valioso, por lo cual se estableció que al trabajar con clientes reales, cada cliente debiera ser atendido por dos grupos desarrollando el mismo proyecto de forma independiente.

Se identificó que muchas de las desviaciones al modelo se debían principalmente a desconocimiento o poco entendimiento de los conceptos planteados en el mismo, como ser el enfoque iterativo incremental y la metodología de Jacobson, Booch y Rumbaugh planteada en [5] debido a la importancia que tiene para la aplicación del modelo. En cuanto a las actividades de Gestión de la Configuración se entiende que para que ésta pueda realizarse en forma similar a las de la industria, por lo que se decidió habilitar la utilización de CVS en forma centralizada con un repositorio para cada grupo.

En cuanto al cumplimiento de la Agenda de Actividades y Entregables definida en el modelo, los grupos en general planteaban que se veían superados por la cantidad de entregables establecida para cada entrega semanal, por lo que se agregó un indicador de la importancia de los Entregables pedidos para cada semana identificado como Prioridad: ALTA, MEDIA, y BAJA, para que los grupos tengan un mayor control sobre la realización o no de los mismos.

En base a los ajustes realizados tanto al modelo Java I como al modelo Java II por los estudiantes como proyecto de grado, se unificaron ambos modelos en el modelo Java que se seguiría en el curso 2001.

Tabla 1: datos de la experiencia 2000

Año 2000					
Grupo	**Integ.**	**Horas**	**LOCs**	**Cliente**	**Modelo**
1	10	1979	9076	Docente	Java I
2	11	1480	13815	Docente	Java I
3	10	1326	3133	Docente	Java I
4	10	s/d	s/d	Docente	Java II
5	11	s/d	s/d	Docente	Java II

5.2 Experiencia 2001

En la edición 2001 se comenzó a trabajar con clientes reales: el grupo Concepción de Sistemas de Información (CSI) del Instituto de Computación, con una herramienta CASE que asiste en el Diseño de Data Warehouses y la empresa de desarrollo de software Netlabs, con el desarrollo de una comunidad virtual

A partir de la evaluación de esta experiencia se incorporó la fase de Transición al cliente, aunque no se les asegura que el producto final pueda ponerse en explotación sin esfuerzo adicional. La inclusión de esta Fase, trajo consigo la incorporación al modelo de la Disciplina de Implantación, conjuntamente con la definición de sus actividades, entregables y roles involucrados. Se consideró oportuno que la evaluación realizada a los clientes sobre los productos finales, incidiera en la evaluación del curso para los grupos de estudiantes, de forma de hacer más real la interacción con los mismos y la preocupación por parte de los grupos del trabajo con el cliente y de la Fase de Transición agregada.

Se volvieron a constatar las observaciones realizadas en la experiencia 2000 sobre la dificultad de tener que estudiar el modelo en paralelo con el desarrollo del proyecto, se decidió incorporar trabajos obligatorios al curso previo teórico Introducción a la Ingeniería de Software, referidos a Casos de Uso, definición de la Arquitectura del Software y Verificación, aspectos en los que se observaban las mayores carencias y dificultades, así como a incluir en el temario del curso la presentación del Modelo Java.

Tabla 2: datos de la experiencia 2001

Año 2001					
Grupo	**Integ.**	**Horas**	**LOCs**	**Cliente**	**Modelo**
1	11	3152	13444	CSI	Java
2	10	2719	19970	CSI	Java
3	11	3919	8907	Netlabs	Java
4	11	3430	12126	Netlabs	Java
5	12	3722	23299	CSI	Java

5.3 Experiencia 2002

En esa edición se contó con dos clientes: el grupo CSI, con tres productos distintos: un sistema que integre dos bases de datos heterogéneas y autónomas utilizando una metodología de integración basada en correspondencias, otro que genere un sistema que permita almacenar, recuperar y modificar los datos generados por una herramienta de diseño de datawarehouse, y otro para la generación de archivos XMI en base a modelos de entrada que representen un esquema estrella. El otro cliente en el año 2002 fue la empresa de desarrollo de software Ideasoft, para quién se realizó un editor gráfico de XML.

En el trabajo con los clientes se presentaron algunas dificultades como malentendidos en la captura de Requerimientos, mala comunicación, priorización de casos de uso (CU) incorrecta. Se observó que en parte esto sucedía porque los grupos no tenían claro cuando debían realizar las validaciones con el cliente ni que productos debían mostrar en cada momento, y también por dificultades planteadas para tener reuniones con el cliente por falta de disponibilidad del mismo. En base a esto, se incluye en el modelo una descripción detallada de las actividades de validación con el cliente estableciendo los productos que se deben mostrar en cada oportunidad, y fijar una cantidad mínima de horas que el cliente debe dedicar al proyecto para reuniones de requerimientos, validaciones y consultas.

Tabla 3 datos de la experiencia 2002

Año 2002					
Grupo	**Integ.**	**Horas**	**LOCs**	**Cliente**	**Modelo**
1	12	3583	14771	IdeaSoft	Java
2	11	4073	21521	IdeaSoft	Java
3	13	3581	14025	CSI	Java
4	13	3716	26500	CSI	Java
5	13	4914	33000	CSI	Java
6	14	5153	26066	CSI	Java
7	14	3200	24000	CSI	Java

5.4 Experiencia 2003

Para la edición 2003, se formalizó el acuerdo cliente-proveedor estableciendo que al inicio de cada proyecto el cliente plantea una idea del mismo, al finalizar la fase Inicial el grupo y el cliente acuerdan un alcance preliminar para el proyecto y al finalizar la fase de Elaboración se establece el alcance final.
En el año 2003 se tuvieron tres clientes distintos: nuevamente la empresa de desarrollo de software Ideasoft, quién planteó construir una herramienta de testing automático de aplicaciones web. Otro de los clientes fue el Centro Latinoamericano de Perinatología y Desarrollo Humano (CLAP), con el producto Sistema Informático del Niño, Escolar y Adolescente. El grupo CSI fue nuevamente cliente, en este caso se realizaron tres productos, en paralelo, siguiendo la extensión Integrado del MP Java, el producto integrado permitía la gestión de metadatas.

Tabla 4 datos de la experiencia 2003

Año 2003					
Grupo	**Integ.**	**Horas**	**LOCs**	**Cliente**	**Modelo**
1	13	4662	14885	IdeaSoft	Java
2	11	4189	10627	IdeaSoft	Java
3	13	4883	31844	CLAP	Java
4	13	4815	29359	CLAP	Java
5	11	4439	15003	CSI	Java Int.
6	12	4049	22389	CSI	Java Int.
7	10	3978	22240	CSI	Java Int.

5.5 Experiencia 2004

En la experiencia 2004 se juntaron los Modelos Java y MoDGX en el Modelo Factor, y se puso en práctica la extensión TLReq, para desarrollo a distancia, en este caso el cliente fué la Unidad Politécnica de Madrid, el producto a realizar a distancia era un sistema de gestión de currículums vitae para un laboratorio académico.

Tabla 5: Datos de la experiencia 2004

Año 2004					
Grupo	**Integ.**	**Horas**	**LOCs**	**Cliente**	**Modelo**
1	11	3526	16609	Cabal	OO Java
2	11	4036	24466	CSI	OO Java
3	11	3123	54000	Gris	OO Java
4	11	4092	33445	CEI	OO Java
5	11	3058	14155	UPM	TLREQ
6	10	3133	35000	UPM	TLREQ
7	10	2845	16293	Link-All	OO Java
8	11	3150	s/d	Correo	OO .Net

El resto de los clientes fueron CABAL uruguay, institución administradora de tarjetas de crédito, el producto era un monitor de transacciones financieras para su visualización y control. Nuevamente el grupo CSI fue cliente con dos productos: una herramienta que permite evaluar la calidad de la información devuelta por un sistema con fuentes de datos heterogéneas, el otro producto fué una aplicación que permite la navegación por una base de datos representada a través de una ontología para el proyecto Link-All., la Oficina de Trabajo del Centro de Estudiantes de la Facultad de Ingeniería con un sistema de inserción laboral, la administración Nacional de Correos de Uruguay con un sistema de venta de productos y servicios, financieros y postales. Por primera vez, el Grupo de ingeniería de Software fue cliente también, esta experiencia fue muy rica para aprender sobre la visión del cliente respecto al proceso y las dificultades de comunicación y coordinación existentes, el producto a construir era un generador web de procesos.

5.6 Experiencia 2005

En la edición 2005 se plantea mejorar la calidad del proceso Factor, principalmente en las disciplinas de calidad y verificación, incluyendo el estudio de la satisfacción del cliente como parte del proceso. Esta edición del proceso se llama Unificado. Se agrega también la extensión SOA (Service Oriented Architecture) que plantea una metodología para desarrollo de aplicaciones orientadas a servicios. Se introduce el uso de la herramienta Bugzilla para el seguimiento de defectos para los equipos de implementación y verificación. Esta edición aún está en curso, los grupos están terminando en su mayoría la Fase de Construcción.

6 Conclusiones y trabajo futuro

Las condiciones en que se aplica el Modelo de Desarrollo de Software OO en el ámbito académico no es directamente trasladable a la Industria, pero presenta algunas similitudes que hacen posible reflexionar sobre la forma en que se construye software y las dificultades que se plantean en su desarrollo. Un desafío planteado es poder obtener mediciones sobre la correspondencia entre el proceso seguido para desarrollar software y la calidad del producto obtenido. Los productos que se obtienen como resultado de los proyectos del curso son prototipos, sin embargo, de las respuestas en los cuestionarios de evaluación del producto por el cliente, creemos que se ha logrado un avance significativo en la satisfacción de los mismos sobre los productos obtenidos, desde las primeras aplicaciones del modelo hasta el presente. El trabajo con clientes ha permitido ajustar aspectos del modelo permitiendo incorporar una visión más real del proyecto por parte de todos los involucrados.

Como trabajo a futuro se plantea: la mejora de la memoria organizacional, de forma de gestionar el conocimiento y no sólo ser un repositorio de información; el estudio de herramientas para apoyar las actividades del proceso en todas sus disciplinas, de forma de automatizar las tareas que sean posibles; realizar un mayor seguimiento durante el proyecto de la satisfacción del cliente; reducir la cantidad de horas que realizan en promedio los estudiantes semanalmente ya que la asignatura tiene 15 horas semanales y tienen un promedio de 25 horas. Para esto creemos que se debe ajustar el proceso en lo que tiene que ver con la negociación del alcance con el cliente y el apoyo en las nuevas tecnologías que tienen una curva de aprendizaje elevada.

Experimentando con procesos, hemos descubierto que al igual que el software requieren mantenimiento, esto lleva tiempo y esfuerzo, así como el conocimiento de quienes lo desarrollan. En este sentido el Grupo de Ingeniería de Software (GrIS) tiene en construcción por parte de dos grupos de estudiantes del curso de este año 2005, dos herramientas para asistir en esta tarea: una que gestione las versiones de los procesos, guardando el histórico de sus cambios y su fundamentación, y genere automáticamente a partir de las mismas la pagina web del proceso, y otra un portal web orientado a asistir en la gestión del curso, donde los grupos puedan hacer la entrega semanal, consultar la agenda, ingresar las horas y realizar las encuestas de evaluación.

7 Referencias

[1] Paulk,M.C. et al, "Capability Maturity Model for Software," V1.1, CMU/SEI-93-TR-024, SEI, 1993.

[2] Grupo de Ingeniería de Software (Gris), Instituto de Computación, Facultad de Ingeniería, Universidad de la República, <http://www.fing.edu.uy/inco/grupos/gris/> [Consulta: diciembre de 2005]

[3] Triñanes J., "TLREQ: Proceso para desarrollo a distancia", en III Jornadas Iberoamericanas de Ingeniería de Software e Ingeniería del Conocimiento (JIISIC'03), Valdivia, Chile, p. 49-52, 2003

[4] Proyecto de Ingeniería de Software, Procesos, Instituto de Computación,Facultad de Ingeniería, Universidad de la República,<http://www.fing.edu.uy/inco/cursos/ingsoft/pis/>[Consulta: diciembre de 2005]

[5] Jacobson, I. Booch, G. Rumbaugh, J."The Unified Software Development Process", Addison Wesley, 1999. ISBN 0-201-57169-2

[6] Software Process Improvement and Capability Determination, V3.3, ISO/IEC-TR-15504, 1999.

[7] Consejo Superior de Informática, "Métrica Versión 3", Madrid, España, 1999-2000. <http://www.map.es/csi/metrica3> [Consulta: diciembre de 2005]

[8] Pressman, R. & Associates, Adaptable Process Model, < http://www.rspa.com/apm> [Consulta: diciembre de 2005]

[9] IBM Rational Unified Process. <http://www-130.ibm.com/developerworks/rational/products/rup/> [Consulta: diciembre de 2005]

[10] Visión General, ARTech, Febrero 2003, <http://www.genexus.com/DOCUM/GeneXus_VG.pdf> [Consulta: diciembre de 2005]

[11] Beck, K. Extreme Programming Explained: Embrace Change, Addison Wesley Professional, 1999, ISBN 201-61641-6

[12] IEEE STD 610, IEEE Standard Glossary of Software Engineering Terminology, 1990.

[13] Delgado, A. Pérez, B. Modelado del proceso de software - Informe final Proyecto de Taller V, Instituto de Computación, Facultad de Ingeniería, Universidad de la República, año 2000