

---

# El Proceso Unificado de Desarrollo de Software.

---

Universidad Tecnológica del Centro de  
Veracruz.

---

Elaborado por: Ana Karina Elotlán Hernández.  
Matricula: 6674.  
Grado y grupo: TICS I 6° A DZ.

---

## Tabla de contenido

Capítulo 1. El proceso unificado: dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental.....	3
1.2. El proceso Unificado está dirigido por casos de uso. ....	3
1.3. El proceso unificado está centrado en la arquitectura. ....	3
1.4. El proceso unificado es iterativo e incremental. ....	3
1.5. La vida del proceso unificado. ....	3
1.5.1. El producto. ....	4
1.5.2. Fases dentro de un ciclo. ....	4
1.6. Un proceso integrado. ....	5
Capítulo 2. Las cuatro “P” en el desarrollo de software: Personas, Proyecto, Producto y Proceso. ....	6
2.1. Las personas son decisivas. ....	6
2.1.1. Los procesos de desarrollo afectan a las personas. ....	6
2.1.2. Los papeles cambiarán. ....	6
2.1.3. Convirtiendo “recursos” en “trabajadores” . ....	6
2.2. Los proyectos construyen el producto. ....	7
2.3. El proceso es más que código. ....	7
2.3.1. ¿Qué es un sistema software? ....	7
2.3.2. Artefactos. ....	7
2.3.3. Un sistema posee una colección de modelos. ....	7
2.3.4. ¿Qué es un modelo? ....	7
2.3.5. Cada modelo es una vista autocontenida del sistema. ....	7
2.3.6. Dentro de un modelo. ....	7
2.3.7. Relaciones entre modelos. ....	8
2.4. El proceso dirige los proyectos. ....	8
2.4.1. El proceso: una plantilla. ....	8
2.4.2. Las actividades relacionadas conforman flujos de trabajo. ....	8
2.4.3. Procesos especializados. ....	8
2.4.4. Méritos del proceso. ....	8
2.5. Las herramientas son esenciales en el proceso. ....	9
2.5.1. Las herramientas influyen en el proceso. ....	9

2.5.2. El proceso dirige las herramientas. ....	9
2.5.3. El equilibrio entre el proceso y las herramientas.....	9
2.5.4. El modelado visual soporta UML.....	9
2.5.5. Las herramientas dan soporte al ciclo de vida completo.....	9
Bibliografía.....	10

## **Capítulo 1. El proceso unificado: dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental.**

La tendencia actual en el software lleva a la construcción de sistemas más grandes y más complejos. Esto es al hecho de que los computadores son más pequeños potentes cada año, y los usuarios, esperan más de ellos. Queremos un software que esté mejor adaptado a nuestras necesidades. Pero esto, a su vez, simplemente hace el software más complejo. También lo queremos más rápido, el tiempo de salida al mercado es otro conductor importante.

### **1.2. El proceso Unificado está dirigido por casos de uso.**

Un proceso de desarrollo de software es el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema software, el proceso unificado es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software.

### **1.3. El proceso unificado está centrado en la arquitectura.**

De manera resumida podemos decir que el arquitecto:

- crea un esquema en borrador de la arquitectura, comenzando por la parte de la arquitectura que no es especificada de los casos de uso. Aunque esta parte de la arquitectura es independiente de los casos de uso, el arquitecto debe poseer una comprensión general de los casos de uso antes de comenzar con la creación del esquema arquitectónico.
- A continuación, el arquitecto trabaja con un subconjunto de los casos de uso especificados, con ellos representa las funciones clave del sistema en desarrollo. Cada caso de uso seleccionado se especifica y se realiza en términos de subsistemas, clases, y componentes.
- A medida que los casos de uso se especifican y maduran, se descubre más de la arquitectura.

Este proceso continua hasta que se considere que la arquitectura es estable.

### **1.4. El proceso unificado es iterativo e incremental.**

Dirigido por casos de uso quiere decir que el proceso de desarrollo sigue un hilo –avanza a través de una serie de flujos de trabajo que parten de los casos de uso. Los casos de uso se especifican, se diseñan y los casos de uso finales son la fuente a partir de la cual los ingenieros de prueba construyen sus casos de prueba.

Beneficios del proceso iterativo controlado:

- Reduce el coste del riesgo a los costes de un solo incremento.
- Reduce el riesgo de no sacar al mercado el producto en el calendario previsto.
- Acelera el ritmo del esfuerzo de desarrollo en su totalidad debido a que los desarrolladores trabajan de manera más eficiente para obtener resultados claros.
- Reconoce una realidad que a menudo se ignora, las necesidades del usuario y sus correspondientes requisitos no pueden definirse completamente al principio.

### **1.5. La vida del proceso unificado.**

Cada ciclo consta de cuatros fases: inicio, elaboración construcción y transición.

### **1.5.1. El producto.**

Los desarrolladores necesitan todas las representaciones del producto software:

- Un modelo de casos de uso, con todos los casos de uso y su relación con los usuarios.
- Un modelo de análisis, con dos propósitos: refinar los casos de uso y establecer la asignación inicial de funcionalidad del sistema a un conjunto de objetos que proporcionan el comportamiento.
- Un modelo de diseño que define la estructura estática del sistema en la forma de subsistemas.
- Un modelo de implementación, que incluye componentes, que representan el código de fuente y la correspondencia de las clases con los componentes.
- Un modelo de despliegue que define los nodos físico (ordenadores) y la correspondencia de los componentes con esos nodos.
- Un modelo de prueba, que describe los casos de prueba que verifican los casos de uso.
- Y, por supuesto, una representación de la arquitectura.

### **1.5.2. Fases dentro de un ciclo.**

Durante la fase de inicio, se desarrolla una descripción del producto final a partir de una buena idea y se presenta el análisis de negocios para el producto. Esencialmente, esta fase responde a las siguientes preguntas:

- ¿Cuáles son las principales funciones del sistema para sus usuarios más importantes?
- ¿Cómo podría ser la arquitectura del sistema?
- ¿Cuál es el plan de proyectos y cuánto costará desarrollar el producto?

Durante la fase de elaboración, se especifican en detalle la mayoría de los casos de uso del producto y se diseña la arquitectura del sistema. La relación entre la arquitectura del sistema y del propio sistema es primordial. Una manera simple de expresarlo es decir que la arquitectura es análoga al esqueleto cubierto por la piel pero con muy poco músculo (el software) entre los huesos y la piel –solo lo necesario para permitir que el esqueleto haga movimientos básicos. Al final de la fase de elaboración, el director de proyecto está en disposición de planificar las actividades y estimar los recursos necesarios para terminar el proyecto.

Durante la fase de construcción se crea el producto se añaden los músculos (software terminado) al esqueleto (la arquitectura). En esta fase, la línea base de la arquitectura crece hasta convertirse en el sistema completo. La descripción evoluciona hasta convertirse en un producto preparado para ser entregado a la comunidad de usuarios.

La fase de transición cubre el periodo durante el cual el producto se convierte en versión beta. En la versión beta un número reducido de usuarios con experiencia prueba el producto e informa de defectos y deficiencias. Los desarrolladores corrigen los problemas e incorporan algunas de las mejoras sugeridas en una versión general dirigida a la totalidad de la comunidad de usuarios.

### **1.6. Un proceso integrado.**

El proceso unificado está basado en componentes y ha establecido un marco de trabajo que integra todas esas diferentes facetas. El marco de trabajo sirve también como paraguas bajo el cual los fabricantes de herramientas y los desarrolladores pueden construir herramientas que soporten la automatización del proceso entero.

## Capítulo 2. Las cuatro “P” en el desarrollo de software: Personas, Proyecto, Producto y Proceso.

- **Personas**: Los principales autores de un proyecto de software son los arquitectos, desarrolladores, ingenieros de prueba, y el personal de gestión que les da soporte, además de los usuarios, clientes, y otros interesados.
- **Proyecto**: Elemento organizativo a través del cual se gestiona el desarrollo de software.
- **Producto**: Artefactos que se crean durante la vida del proyecto.
- **Proceso**: Un proceso de ingeniería de software es una definición del conjunto completo de actividades necesarias para transformar los requisitos de usuario en un producto.

### 2.1. Las personas son decisivas.

Las personas financian el producto, lo planifican, lo desarrollan, lo gestionan, lo prueban, lo utilizan y se benefician de él. El proceso que guía este desarrollo debe funcionar bien para las personas que lo utilizan.

#### 2.1.1. Los procesos de desarrollo afectan a las personas.

Aquí se describen los conceptos

- Viabilidad del proyecto: Una aproximación iterativa en el desarrollo permite juzgar pronto la viabilidad del proyecto.
- Gestión del riesgo: La exploración de los riesgos significativos en las primeras fases atenúa este problema.
- Estructura de los equipos. Una buena arquitectura, o interfaces bien definidas entre subsistemas y componentes hace posible una división del esfuerzo de este tipo.
- Planificación del proyecto: Las técnicas que se utilizan en las fases de inicio y de elaboración permiten a los desarrolladores tener una buena noción de cuál debería ser el resultado del proyecto.
- Facilidad de comprensión del proyecto: La descripción de la arquitectura proporciona una visión general para cualquiera que se encuentre implicado en el proyecto.
- Sensación de cumplimiento: La retroalimentación frecuente y las conclusiones obtenidas aceleran el ritmo de trabajo.

#### 2.1.2. Los papeles cambiarán.

Debido a que los cambios del mundo real seguirán sucediendo durante todos estos ciclos de vida, los sistemas tendrán que diseñarse de un modo que les permita crecer durante largos periodos de tiempo

En los años venideros, la mayoría de los informáticos van a empezar a trabajar más estrechamente con sus objetivos, y serán capaces de desarrollar software más complejo gracias a un proceso automatizado y a los componentes reutilizables.

#### 2.1.3. Convirtiendo “recursos” en “trabajadores”.

Un trabajador es un papel que un individuo puede desempeñar en el desarrollo de software.

Cada trabajador es responsable de un conjunto completo de actividades, como las actividades necesarias para el diseño de un subsistema. Necesitan comprender cuáles son sus roles en relación con los de otros trabajadores. Las herramientas no sólo deben ayudar a los trabajadores a llevar a cabo sus propias actividades, también deben aislarlos de la información que no les sea relevante.

## **2.2. Los proyectos construyen el producto.**

- Una secuencia de cambio: Los proyectos de desarrollo de sistemas obtienen productos como resultados, y el camino hasta obtenerlos es una serie de cambios.
- Una serie de iteraciones: Dentro de cada fase de un ciclo, los trabajadores llevan a cabo las actividades de la fase a través de una serie de iteraciones.
- Un patrón organizado: Indica los tipos de trabajadores que el proyecto necesita y los artefactos por los cuales hay que trabajar.

## **2.3. El proceso es más que código.**

El producto que se obtiene es un sistema software, hace referencia al sistema entero, y no sólo al código que se entrega.

### **2.3.1. ¿Qué es un sistema software?**

Un sistema de software son todos los artefactos que se necesitan para representarlo en una forma comprensible por la máquinas u hombres.

### **2.3.2. Artefactos.**

Es un término general para cualquier tipo de información creada, producida, cambiada o utilizada por los trabajadores en el desarrollo del sistema.

Existen dos tipos de artefactos: artefactos de ingeniería y artefactos de gestión.

### **2.3.3. Un sistema posee una colección de modelos.**

La construcción de un sistema es por tanto un proceso de construcción de modelos, utilizando distintos modelos para describir todas las perspectivas diferentes del sistema. El proceso unificado proporciona un conjunto de modelos cuidadosamente seleccionado con el cual comenzar.

### **2.3.4. ¿Qué es un modelo?**

Un modelo es una abstracción del sistema, especificación del sistema modelado desde un cierto punto de vista y en un determinado nivel de abstracción.

### **2.3.5. Cada modelo es una vista autocontenida del sistema.**

Un modelo es una abstracción semánticamente cerrada del sistema. Es una vista autocontenida en el sentido de que un usuario de un modelo no necesita para interpretarlo más información.

La mayoría de los modelos de ingeniería se definen mediante un subconjunto de UML cuidadosamente seleccionado.

### **2.3.6. Dentro de un modelo.**

Un modelo siempre identifica el sistema que está modelando. Este elemento del sistema es por tanto el contenedor de los demás elementos.



### **2.3.7. Relaciones entre modelos.**

Un sistema contiene todas las relaciones y restricciones entre elementos incluidos en diferentes modelos. Por tanto un sistema no es solo la colección de sus modelos sino que contiene todas las relaciones entre ellos.

Existen trazas entre las colaboraciones del modelo de diseño y las colaboraciones en el modelo de análisis, y entre los componentes en el modelo de implementación y los subsistemas en el modelo de diseño.

### **2.4. El proceso dirige los proyectos.**

Proceso es un término demasiado utilizado. Proceso unificado, se refiere a los procesos “de negocio” claves en una empresa de desarrollo de software.

#### **2.4.1. El proceso: una plantilla.**

En el Proceso unificado hace referencia a un contexto que sirve como plantilla que puede reutilizarse para crear instancias de ellas, este proceso no cubre solamente el primer ciclo de desarrollo sino también los posteriores más comunes.

Un proceso es una definición de un conjunto de actividades, no su ejecución.

#### **2.4.2. Las actividades relacionadas conforman flujos de trabajo.**

Describimos el proceso entero en partes llamadas flujos de trabajo, en términos UML, un flujo de trabajo es un estereotipo de colaboración, en el cual los trabajadores y los artefactos son los participantes.

#### **2.4.3. Procesos especializados.**

Los factores principales que influyen en cómo se diferenciará el proceso son:

- Factores organizativos: La estructura organizativa, la cultura de la empresa, la organización y la gestión del proyecto.
- Factores del dominio: El dominio de la aplicación, procesos de negocio que se deben soportar.
- Factores del ciclo de vida: El tiempo de salida al mercado, el tiempo de vida esperando del software.
- Factores técnicos: Lenguaje de programación, herramientas de desarrollo, base de datos, marcos de trabajo y arquitecturas “estándar” subyacentes, comunicaciones y distribución.

#### **2.4.4. Méritos del proceso.**

- Todo el mundo en el equipo de desarrollo puede comprender lo que tiene que hacer para desarrollar el producto.
- Los desarrolladores pueden comprender mejor lo que los otros están haciendo.
- Los supervisores y directores, incluso los que no entienden el código, pueden comprender lo que los desarrolladores están haciendo gracias a los esquemas de la arquitectura.
- Los desarrolladores, los supervisores y los directores pueden cambiar de proyecto o de división sin tener que aprender un nuevo proceso.
- La información puede estandarizarse dentro de la empresa, y puede obtenerse de compañeros o de cursos breves.
- El devenir del desarrollo del software es repetible, puede planificarse y estimarse en coste con suficiente exactitud como para cumplir las expectativas.

## **2.5. Las herramientas son esenciales en el proceso.**

Soportan los procesos de desarrollo de software modernos. El proceso y las herramientas vienen en el mismo paquete.

### **2.5.1. Las herramientas influyen en el proceso.**

Las herramientas son buenas para automatizar procesos repetitivos y para guiarnos a lo largo de un camino de desarrollo concreto.

### **2.5.2. El proceso dirige las herramientas.**

Las herramientas que implementan un proceso automatizado deben ser fáciles de usar, la herramienta debe ser sencilla de comprender y manejar para los trabajadores, debe proporcionar un incremento de productividad sustancial.

### **2.5.3. El equilibrio entre el proceso y las herramientas.**

Debe haber un equilibrio entre proceso y herramientas.

El desarrollo del proceso y su soporte por herramientas debe tener lugar de manera simultánea y en cada versión del proceso debe haber también una versión de las herramientas.

### **2.5.4. El modelado visual soporta UML.**

UML define reglas sintácticas que especifican cómo combinar elementos del lenguaje, la herramienta debe ser capaz de garantizar que se cumplen esas reglas. La herramienta de modelado debe incorporar más que conocimientos de UML.

UML incluye un cierto número de reglas semánticas que también requieren soporte. Estas herramientas pueden incluirse en la herramienta de modelado.

### **2.5.5. Las herramientas dan soporte al ciclo de vida completo.**

- Gestión de requisitos: Se utilizan para almacenar, examinar, revisar, hacer el seguimiento y navegar por los diferentes requisitos de un proyecto software.
- Modelado visual.
- Herramientas de programación: Proporciona una gama de herramientas, incluyendo editores, compiladores, depuradores, detectores de errores y analizadores de rendimiento.
- Aseguramiento de calidad.

## **Bibliografía.**

Jacobson I., Booch G., Rumbaugh J. (2000). *El proceso unificado de desarrollo de Software*. Madrid. Addison Wesley.