

## UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE

**Nombres:** Kevin Cañola, Cristhian Marcalla, Eduardo Tasiguano, Esteban Lugmaña.

**NRC:** 29 022

**Fecha:** 17 nov. 25

### Taller: Análisis Crítico del Proceso Unificado Racional (RUP)

#### Objetivo del Taller

Fomentar el pensamiento crítico en los estudiantes a través del análisis, discusión y aplicación del RUP como metodología de desarrollo de software, reconociendo sus fases, disciplinas y adaptabilidad a diferentes contextos de proyectos.

#### Actividad 1: Exploración Guiada

##### Instrucciones:

1. Lee el siguiente fragmento del Capítulo IX sobre RUP. 2. Extrae: - Las **4 fases** principales del RUP. - 2 diferencias entre **disciplinas de desarrollo** y **disciplinas de soporte**. - Ventajas del enfoque iterativo e incremental.

##### 1. Fases:

###### 1. Concepción (Inicio)

- Se establece el caso y se identifican entidades externas que interactúan con el sistema.

###### 2. Elaboración

- Se comprende el problema del dominio, se planifican actividades, responsables y recursos.

###### 3. Construcción

- Se diseña, programa y prueba el producto; se entrega un software funcional y documentación para el usuario.

###### 4. Transición

- Se pone el software en un ambiente real, incluye capacitación, entrenamiento y pruebas de funcionamiento.

##### 2. diferencias

Aspecto	Disciplinas de Desarrollo	Disciplinas de Soporte
Propósito principal	Crear y evolucionar el producto software.	Gestionar, controlar y facilitar el proceso de desarrollo.

Aspecto	Disciplinas de Desarrollo	Disciplinas de Soporte
<b>Tipo de actividades que realizan</b>	Modelado del negocio, requerimientos, análisis y diseño, codificación, pruebas e instalación.	Administración de la configuración y cambios, administración de proyectos y administración del ambiente.
<b>Resultado esperado</b>	Artefactos del software: modelos UML, código, prototipos, versiones funcionales.	Instrumentos de control y gestión: planes, control de versiones, manejo de recursos y entorno.
<b>Relación con el producto</b>	Directamente construyen y mejoran el software.	No construyen software, apoyan y organizan el proceso.

### ***3. Ventajas del enfoque iterativo e incremental***

Basándose en lo que explica el capítulo:

1. **Permite entregar prototipos funcionales en cada ciclo**  
Cada iteración entrega “un prototipo funcional del producto”, lo que ayuda a validar requisitos y recibir feedback temprano.
2. **Cada fase se ejecuta de manera iterativa, no lineal**  
RUP evita un enfoque rígido tipo cascada; cada fase se repite mejorando continuamente el producto.
3. **Facilita manejar complejidad y riesgos**  
Gracias a las iteraciones, la comprensión del problema y la calidad del software mejoran progresivamente.

### ***Actividad 2: Debate Crítico***

#### **1. ¿Aplicarías RUP en este caso? ¿Por qué?**

El RUP es muy robusto y ofrece una estructura clara para proyectos complejos, pero también es pesado, requiere experiencia previa, documentación abundante y roles bien definidos. En una empresa con recursos limitados y sin conocimiento previo en RUP, su adopción total podría ralentizar el proyecto.

Por eso, sería mejor aplicar un RUP simplificado, centrándose en sus fases y disciplinas esenciales, usando iteraciones cortas y priorizando los artefactos indispensables.

#### **Razones a favor:**

- Permite controlar riesgos desde el inicio.
- Es iterativo → facilita entregas parciales en un proyecto de solo 6 meses.

- Maneja bien requisitos cambiantes.

#### **Razones en contra:**

- Curva de aprendizaje alta.
- Puede generar mucha documentación.
- Su implementación completa consumiría más tiempo que el disponible.

## **2. ¿Qué fases o disciplinas serían prioritarias?**

### **Fases prioritarias**

En este proyecto, lo más importante sería:

#### **1. Inicio**

- Identificar objetivos del sistema académico.
- Definir alcance realista para 6 meses.
- Identificar riesgos principales.
- Estimar tiempo y recursos.

#### **2. Elaboración – Fase crítica**

- Modelado de la arquitectura.
- Construcción del prototipo arquitectónico.
- Identificar casos de uso principales (matrículas, notas, usuarios, reportes).
- Validar que la arquitectura es viable con poco personal.

#### **3. Construcción – Iterativa**

- Desarrollo por módulos:
  - Gestión de estudiantes
  - Gestión de cursos
  - Matrículas
  - Calificaciones
- Pruebas en cada iteración.

#### **4. Transición – versión mínima**

- Capacitar al personal de la institución.
- Corrección de errores críticos.
- Despliegue.

## **Disciplinas RUP prioritarias**

### **Desarrollo:**

1. **Modelado de negocio (leve)**  
Para entender el proceso académico actual.
2. **Requisitos**  
Definir casos de uso mínimos e indispensables.
3. **Análisis y diseño**  
Crear una arquitectura ligera y escalable.
4. **Implementación**  
Desarrollo en iteraciones pequeñas.
5. **Pruebas**  
Validación funcional en cada iteración.

### **Soporte:**

1. **Gestión de configuración**  
Control de versiones del software.
2. **Gestión de proyectos**  
Planificación por iteraciones.

## **3. ¿Qué riesgos podrían surgir al usar RUP?**

### **1. Sobrecarga documental**

RUP produce muchos artefactos, lo que puede consumir tiempo valioso.

### **2. Falta de experiencia del equipo**

Sin práctica previa, pueden cometer errores en:

- Casos de uso
- Diagramas UML
- Definición de arquitectura
- Estimaciones

### **3. Iteraciones mal definidas**

El proyecto puede perder foco si las iteraciones no están bien gestionadas.

### **4. Exceso de formalidad**

La empresa puede dedicar más tiempo al proceso que al producto si intenta seguir RUP al pie de la letra.

### **5. Riesgo de no cumplir los 6 meses**

Si la fase de elaboración se extiende demasiado, no habrá tiempo suficiente para la construcción.

## CONCLUSIÓN

Aunque RUP completo no es la mejor opción por el tiempo limitado y la falta de experiencia del equipo, una versión adaptada del proceso puede ser útil. Si se enfocan solo las actividades esenciales, como definir el alcance, aclarar los requisitos y establecer una arquitectura estable, el proyecto puede avanzar de manera más organizada y con menos riesgos. Bien aplicado y sin exceso de documentación, RUP puede aportar orden y control sin sobrecargar al equipo.

## BIBLIOGRAFÍA

- Clayton, M. (2024, 7 marzo). What is the Rational Unified Process – RUP? OnlinePMCourses. <https://onlinepmcourses.com/what-is-the-rational-unified-process-rup/>
- Skrobak, G. (2015). El Proceso Unificado de desarrollo de software. [www.academia.edu.  
https://www.academia.edu/11946867/El\\_Proceso\\_Unificado\\_de\\_desarrollo\\_de\\_software](https://www.academia.edu/11946867/El_Proceso_Unificado_de_desarrollo_de_software)
- E, A. A. D. (2007). RATIONAL UNIFIED PROCESS (RUP) EN EL PROCESO DE DESARROLLO DE SOFTWARE. <https://dspace.ucacue.edu.ec/items/3496660f-aab3-4967-bc26-26525c5a1b4d>