**CONCEPTO**

El Proceso Unificado Rational (RUP) es el proceso de ingeniería de software, cuyo objetivo es producir software de alta calidad, es decir, que cumpla con los requerimientos de los usuarios dentro de los márgenes de la planificación y presupuestos establecidos. El RUP, cubre todo el ciclo de vida de desarrollo de software, el propósito es asegurar la producción de software, es decir, que colme las expectativas y exigencias del usuario actual, entregado en el tiempo previsto, con la calidad esperada, que se maneje dentro del presupuesto-costo calculado y que cumpla con los requisitos establecidos en la definición del proyecto de construcción del software.

El RUP puede integrar todos los aspectos a tener en cuenta durante el ciclo de desarrollo del software con el objetivo de hacer tangibles todo tipo de proyectos sin interesar su envergadura.

**IMPORTANCIA PROCESO RUP**

* Permite dar solución a los exigentes requerimientos de los usuarios actuales, cada vez más exigentes, debido a los constantes cambios que la misma sociedad y competencias en el mercado exigen.
* Permite obtener los requerimientos y organizarlos, documentar los requerimientos de funcionalidad y restricciones, documentar decisiones, captarlas y por último comunicar los requerimientos del negocio.
* Permite capturar varias de las mejores prácticas en el desarrollo moderno de software de forma que sea aplicable en un amplio rango de proyectos y organizaciones.
* Es una guía de cómo utilizar de manera efectiva el UML.
* La técnica de modelado UML, no se utiliza únicamente para efectos de documentación, gracias al proceso RUP, el UML está presente en todas las fases y etapas establecidas por RUP, con UML cada uno de los roles participantes en el proceso de desarrollo de software pueden expresar su trabajo en términos de diagramas.

Los analistas, ingenieros, arquitectos de software, revisores de casos de uso, etc., Utilizan los diagramas para mostrar el detalle de la construcción del software.

* Provee a cada miembro de equipo el fácil acceso a una base de conocimiento con guías, plantillas y herramientas para todas las actividades críticas de desarrollo.
* Crea y mantiene modelos, en lugar de enfocarse en la producción de gran cantidad de papeles de documentación.
* Permite que todos los miembros del equipo compartan: Conocimiento base, el proceso, la visión de cómo desarrollar software y el lenguaje de modelado.
* Permite la verificación de la calidad del software, mediante las siguientes actividades:
* Crea pruebas para cada escenario (casos de uso), asegurando que todos los requerimientos están apropiadamente implementados.
* Verifica la calidad del software con respecto a los requerimientos basados en la confiabilidad, funcionalidad, desempeño de la aplicación y del sistema.
* Prueba cada iteración.
* El proceso de Pruebas, sujeto también al modelo iterativo e incremental, permite que cada caso de uso que NO cumpla con el control de calidad pueda corregirse e implementarse en el momento indicado ya que la implementación de la solución obviamente buena, puede no ser la solución idónea si no es implementado en el momento justo.

**PRINCIPIOS DE RUP**

* ***GUIADO POR CASOS DE USO***

La razón de ser del sistema de software es servir a los usuarios, ya sean humanos u otros sistemas. El caso de uso o fragmento funcional del sistema, es la facilidad que el software provee a los actores (personas, software o hardware) que utilizan o son utilizados por la plataforma de información del sistema.

Los casos de uso, son importantes ya que definen el comportamiento del futuro sistema, los casos de uso NO son parte de la orientación a objetos tradicional, pero entienden su funcionalidad, cada vez más clara y precisa a medida que se evoluciona; otros métodos orientados a objetos usan los casos de uso como guía pero con nombres diferentes.

Los casos de uso juegan un papel importante en cuatro etapas de los procesos generales del RUP:

* Requisitos
* Análisis-Diseño
* Codificación
* Prueba.
  + El modelo de casos de uso es el resultado del análisis en la etapa de “Requisitos ò Análisis de Requerimientos”, en esta temprana etapa se necesitan a los casos de uso para conocer que hará el software desde el punto de vista del usuario, los casos de uso constituyen un concepto importante y fundamental, deben ser aceptados por el cliente y el grupo desarrollador.
  + En la segunda etapa “Análisis-Diseño”, los casos de uso son ejecutados en el modelo de diseño, se crea realizaciones de casos de uso, que describa como los casos de uso son realizados en términos de interacción de objetos en el modelo de diseño, este modelo describe en términos de diseño de objetos las diferentes partes de la implementación del sistema y cómo deben actuar o funcionar las partes.
  + En la tercera etapa “Implementación”, el modelo de diseño es la especificación de la implementación, porque los casos de uso realizados en el modelo de diseño, son implementados en términos de diseño de clases.
  + Durante la cuarta etapa “Pruebas”, los casos del uso constituyen la base para identificar la prueba de casos de uso y la prueba de procedimientos, el sistema pasará el control de calidad y la etapa de pruebas solo si todos los casos de uso especificados y desarrollados en etapas anteriores son parte funcional del sistema final.
  + Al utilizar el modelamiento de negocio se hallan los diversos procesos empresariales de la organización convirtiéndolas en casos de uso de negocio, posteriormente en casos de uso del sistema según el alcance del proyecto.
* ***CENTRADO EN LA ARQUITECTURA***

El RUP, enfatiza la construcción de sistemas software robusto, respetando la arquitectura de construcción, ello disminuye el reinicio del software, aumentado la reutilización y facilita el mantenimiento futuro del mismo.

La arquitectura se utiliza para planificar y administrar el desarrollo del software teniendo en cuenta la reutilización de sus componentes.

La arquitectura involucra los elementos más significativos del sistema y está influenciada entre otros por plataformas software, sistemas operativos, gestores de base de datos, protocolos de comunicación, etc.

El enfoque de iteraciones tempranas, definido con mayor énfasis en la fase de elaboración es producir y validar una arquitectura de software, que el ciclo de desarrollo inicial toma la forma de un prototipo arquitectónico ejecutable el cual evoluciona gradualmente para convertirse en un sistema final en las últimas iteraciones.

* ***RESPETA EL MODELO ITERATIVO E INCREMENTAL***

El RUP es un proceso iterativo e incremental, el cual permite entender el problema a través de sucesivos refinamientos e incrementar la solución efectiva mediante múltiples iteraciones, este acercamiento brinda la mejor opción en acomodar nuevos requerimientos o cambio de tácticas en los objetivos del negocio y continuar con el proyecto identificado, resolviendo riesgos de manera oportuna.

El RUP es un proceso controlado, la característica iterativa solo es posible al menos a través de una cuidadosa administración de requerimientos y control de cambios, asegurando así la comprensión de la funcionalidad del software en el momento adecuado considerando la calidad prevista, además permite el control de la entrega del proyecto dentro del tiempo establecido.

Para hacer más manejable un proyecto se recomienda dividirlo en ciclos, para cada ciclo se establecen fases de referencia, cada una de las cuales debe ser considerada como mini-proyecto cuyo núcleo fundamental está constituido por una o más iteraciones de las actividades principales básicas de cualquier proceso de desarrollo.

La característica iterativa del RUP, permite:

* + El entendimiento incremental del problema a través de refinamientos sucesivos.
  + Habilitar la fácil retroalimentación del usuario.
  + Establecer metas específicas que permiten al equipo de desarrollo mantener su atención en producir resultados.
  + La medición del progreso es conforme avanzan las implementaciones.
* ***DIRECCIÓN O ADMINISTRACIÓN DE LOS REQUISITOS***

Es el acercamiento sistemático a encontrar resultados, mientras se documenta, se organiza y se rastrea los requisitos de un sistema.

Formalmente es el establecimiento del acuerdo entre los clientes y el grupo del proyecto para administrar los cambios de requerimientos en el sistema. Los puntos clave en el manejo de requisitos son el mantenimiento de una visión clara de los requisitos en conjunto con los atributos aplicables y la proyección a otros requisitos y/o artefactos del proyecto.

• Los requerimientos no son fáciles de expresar claramente en palabras.

• La abundancia de requerimientos puede ser difícil de manejar por ende de controlar.

• Los requerimientos cambian con mucha frecuencia.

• Existen diversos tipos de requerimientos y diferentes niveles de detalle.

• Los requisitos no siempre son obvios y tienen diferentes fuentes.

Se debe tener en cuenta las siguientes habilidades para lograr el éxito aun con las dificultades que pueden presentar los requisitos:

• El análisis y entendimiento del problema.

• Comprender las necesidades de cada uno de los involucrados en el proyecto.

• Definir claramente el sistema en base a casos de uso.

• Definir claramente el alcance del proyecto.

• Refinar constantemente la definición del sistema.

• Realizar el seguimiento y control a los requisitos cambiantes.

* ***PROCESO CONFIGURABLE***

El Proceso Unificado Rational (RUP), es bastante general y completo, puede ser usado en muchas organizaciones de software. En muchas circunstancias, este proceso de ingeniería de software necesitará ser modificado, ajustado, extendido y entallado para acomodarse a las características específicas, circunstancias, entorno cultural, organizacional y político de la organización que lo adopta. Los elementos del proceso de ingeniería de software que probablemente serán modificados, personalizados, agregados o suprimidos son los siguientes:

* Artefactos
* Actividades
* Flujos de trabajo
* Obreros
* ***TECNICAS DE MODELAMIENTO VISUAL***

El RUP, al utilizar diferentes técnicas de modelamiento visual, permite:

* La captura de la estructura, comportamiento de arquitecturas y componentes.
* Mostrar como encajan de forma conjunta los elementos del sistema.
* Mantener la consistencia entre un diseño y su implementación.
* Promover la comunicación no ambigua.
* ***DESARROLLO BASADO EN COMPONENTES***

Gracias a la propiedad de herencia, adoptado de la Metodología Orientada a Objetos, el proceso RUP, permite el desarrollo de software basado en componentes, el cual brinda ventajas importantes como:

* Permite enfocarse en el pronto desarrollo de una arquitectura ejecutable robusta.
* Es intuitivamente comprensible.
* Promueve la reutilización más efectiva de software.
* Permite la construcción rápida de software. Si tienes gran cantidad de componentes reutilizables se puede finalizar el proyecto informático en un tiempo realmente corto.
* Es derivada a partir de los casos de uso más importantes.

***ESTRUCTURA DEL RUP***

El proceso puede ser descrito en dos dimensiones o ejes:

**Eje horizontal:** Representa el tiempo y es considerado el eje de los aspectos dinámicos del proceso. Indica las características del ciclo de vida del proceso expresado en términos de fases, iteraciones e hitos. Se puede observar en la Figura que RUP consta de cuatro fases: Inicio, Elaboración, Construcción y Transición.

**Eje vertical:** Representa los aspectos estáticos del proceso. Describe el proceso en términos de componentes de proceso, disciplinas, flujos de trabajo, actividades, artefactos y roles.



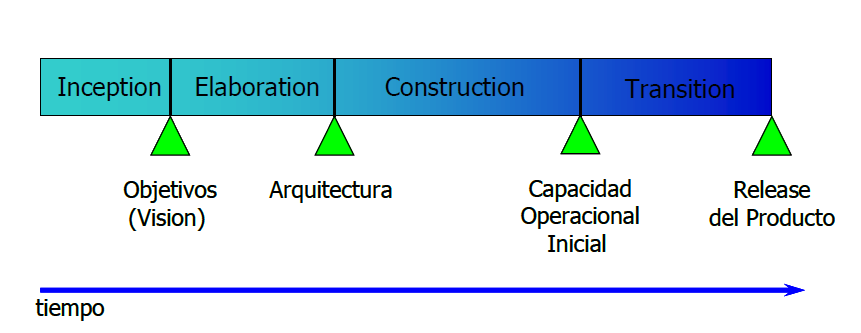
RUP se repite a lo largo de una serie de ciclos que constituyen la vida de un producto. Cada ciclo concluye con una generación del producto para los clientes. Cada ciclo consta de cuatro fases:

1. Inicio
2. Elaboración
3. Construcción
4. Transición.

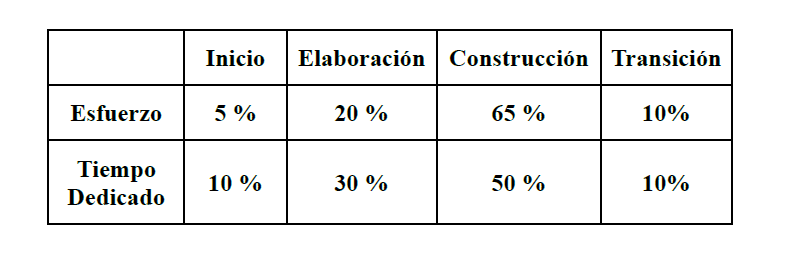
Cada fase se subdivide a la vez en iteraciones, el número de iteraciones en cada fase es variable. Cada fase se concluye con un hito bien definido, un punto en el tiempo en el cual se deben tomar ciertas decisiones críticas y alcanzar las metas clave antes de pasar a la siguiente fase.

Los hitos para cada una de las fases son:

* Inicio -> Objetivo del ciclo de vida.
* Elaboración –> Arquitectura del ciclo de vida.
* Construcción –> Capacidad operacional inicial.
* Transición –> Producto entregado.



La duración y esfuerzo dedicado en cada fase es variable dependiendo de las características del proyecto. Sin embargo, las siguientes figuras ilustra porcentajes frecuentes al esfuerzo señalado, y otra ilustra una distribución típica de recursos humanos necesarios a lo largo del proyecto.





***FASES***

* ***GESTACIÓN O CONCEPCIÓN***

Esta fase permite el establecimiento de los objetivos y el plan del proyecto al definir su alcance.

El propósito es establecer los casos de uso de negocio para el nuevo sistema o para alguna actualización importante del sistema existente.

En esta etapa se establece la visión general de los requerimientos del proyecto y de los requerimientos principales para la construcción del software, definiendo el modelo inicial de casos de uso y el modelo del dominio.

Se realiza la evaluación inicial de riesgos y la estimación de los recursos requeridos, para minimizar los riesgos ó evitar que los riesgos se conviertan en problemas.

Esta etapa es ideal para definir el glosario de conceptos, políticas de creación y nombramiento de artefactos.

En esta etapa se realiza lo siguiente:

* Captura de requisitos.
* Diagramas de casos de usos.
* Diagrama de clase de análisis.
* Diagrama de colaboración.
* Diagrama de paquetes.
* ***PREPARACIÓN O ELABORACIÓN***

Permite la definición de la arquitectura, desarrollo del plan del proyecto y la especificación de características del sistema.

Esta etapa permite definir la funcionalidad del software a construir, al clasificar y priorizar los casos de uso del sistema.

Se desarrolla la programación lógica expresada en diagramas, se realiza el análisis y diseño de la base de datos, pasando por el modelo conceptual, el modelo lógico y el modelo físico de la base de datos.

En esta etapa se realiza lo siguiente:

* Diagrama de clase de diseño
* Diagrama de capas
* Diseño de la base de datos.
* Diseño de la interfaz.
* Diagrama de secuencia.
* Diagrama de componentes.

**1.1 CONSTRUCCIÓN**

En esta fase se desarrolla el código final, se construye el producto final.

El propósito de esta fase es desarrollar incrementalmente el producto software completo el cual estará listo para ser transferido al usuario:

Se producen los siguientes artefactos:

• Después de realizar el proceso de ingeniería reversa, se realiza el modelo completo de diseño y casos de uso, producto del código de la implementación

• Liberaciones de productos ejecutables de funcionalidad incremental (versiones, prototipos)

• Documentación técnica del sistema

• Manuales de usuario• Se obtiene la versión “beta” del producto

* 1. **TRANSICIÓN**

Se realiza la transición del producto en el entorno usuario**.** Esta fase, está dedicada a establecer los lineamientos de performance, logrando así cubrir las expectativas de usuario.

Se obtienen los siguientes artefactos:

• Producción de ejecutables de producto

• Modelo de componentes al realizar la compilación y despliegue de componentes en base a la *ingeniería reversa*

• Modelo de diseño actualizado en base a la Ingeniería reversa

• “Pruebas beta” del software para validar el nuevo sistema versus las expectativas del usuario

• Manuales de usuario actualizados

• Documentación de desarrollo actualizada

• Se realiza el proceso de retroalimentación desde el punto de vista del usuario referente a la reciente implementación. Se contesta a las siguientes preguntas ¿el usuario está satisfecho?, ¿los gastos reales de los recursos versus gastos previstos son aceptables?

Uno de los puntos de importancia en esta fase es el desarrollar el plan de soporte y mantenimiento para el sistema de información que acaba de ser puesto en producción, se define cada cuanto tiempo se realizará el mantenimiento, características del equipo de mantenimiento, procesos de supervisión, políticas a seguir en el proceso de copia de seguridad y las políticas de registro a los nuevos usuarios.

1. **Etapas del RUP**

**2.1. DISCIPLINAS CENTRALES**

**2.1.1. MODELO DE NEGOCIO**

Analiza la organización en términos de procesos y las personas u organizaciones que influencian o participan en él, de forma directa o indirecta. El modelo de negocio sirve para determinar cuál es el problema de la organización. Presenta dos etapas:

• ***MODELO DE CASOS DE USO DE NEGOCIO***

Este modelo muestra la relación existente entre un actor de negocio externo ó interno y al caso de uso de negocio, desde el punto de vista general.

• ***MODELO DE OBJETOS DE NEGOCIO***

En esta etapa se define el detalle del negocio en términos de procesos empresariales; el caso de uso de negocio definido en la etapa anterior pasa por el proceso de realización, donde utilizando diferentes vistas del UML como: diagrama de casos de uso (realización del caso de uso), diagrama de clase, diagramas de secuencia, diagramas de colaboración e incluso el diagrama de actividades se llega al detalle de funcionalidad del proceso empresarial que se analiza.

**2.1.2. FUNCIONALIDAD**

En esta etapa se define ¿qué hace el sistema?, para responder la interrogante anterior se realiza diversos procesos para el análisis de los requerimientos, logrando definir cuáles serán las opciones del menú principal del sistema incluyendo cada uno de las sub opciones incluso definir las interfaces del sistema final.

**2.1.3. ANÁLISIS**

Esta etapa está dirigida al análisis de la información obtenida en el negocio, después de haber definido la funcionalidad del software que se construye en la etapa anterior, es necesario definir como se realizará la implementación en base a todos los requerimientos establecidos. El Proceso Unificado Rational, propone los denominados, clasificadores de análisis, para realizar la programación lógica; este muestra el detalle de cómo se realizará los procesos de funcionalidad del software final.

Ejemplo: ¿cómo se inserta un registro a una tabla independiente?

Para el desarrollo de la interrogante anterior es necesario conocer las siguientes entradas:

• En qué parte del menú principal se encuentra la opción de inserción del nuevo registro.

• Cuántas y cuáles son las interfaces que participan en el proceso de inserción del registro

• Cuál es la tabla de la base de datos, obviamente, debemos definir la estructura de la futura tabla, que aún no existe.

• Por último se propone un modelo utilizando los clasificadores de análisis y artefactos del UML mostrando la relación lógica de participación e interacción entre cada uno de los artefactos que participan para lograr insertar el registro en una tabla independiente.

**2.1.4. DISEÑO**

Esta etapa causa duda y controversia, entre muchos autores de libros o artículos en la web, en más del 60% del material bibliográfico investigado para la presente edición, existe dudas con respecto a esta etapa, para muchos autores esta es la etapa de implementación lógica del software, hay algunos que pretenden hacer una comparación con la metodología estructurada y su típico modelo entidad/relación (E/R) para la construcción de la base de datos. Es lamentable que las justificaciones sean sólo teóricas.

La programación lógica ya fue definida en la etapa de análisis, en la etapa de diseño nos dedicamos a la construcción de la base de datos relacional / objeto.

Es importante mencionar que NO existe comunión entre el modelo **E/R** y el modelo de **Objetos**. Es imposible compararlos ya que tienen puntos de partida y consecuencias diferentes. En diseño tenemos que tomar en cuenta la información proveniente de etapas anteriores, se inicia una de las actividades más importantes en el proceso de construcción de software, “el análisis y diseño de la base de datos”, y la primera propuesta del modelo de diseño del futuro software.

Realizaremos el modelo orientado a objetos, definido por etapas iterativas de desarrollo desde el modelo conceptual, pasando por el modelo lógico hasta llegar al modelo físico que es el modelo de base de datos relacional – objeto propiamente dicho.

El objetivo del modelo orientado a objetos es la construcción de una base de datos relacional – objeto que cumpla con todos los criterios de performance y calidad.

Los criterios de calidad de la base son los siguientes: la base de datos No debe permitir nulos, ni redundancia logrando la homogeneidad, sin pasar por el tedioso método de normalización, el cual es un término no existente en el proceso RUP.

**2.1.5. IMPLEMENTACIÓN**

En esta etapa se administra la generación de archivos, empezamos la codificación para producir el software final, se debe tener en cuenta los artefactos producidos en las anteriores etapas, sobre todo el modelo de la base de datos relacional – objeto (clases con el estereotipo tabla *relacional - objeto*, estructura, relaciones entre clases, las llaves primarias y los campos foráneos) y el *modelo de análisis* (programación lógica en términos de diagramas).

Ya terminada la etapa de implementación, realizamos el proceso de *ingeniería reversa* para actualizar para primera propuesta del diseño, este proceso es útil para garantizar el cumplimiento de todos los requerimientos del usuario.

**2.1.6. CERTIFICACIÓN**

Esta etapa analizamos los prototipos, por cada caso de uso. Gracias a la característica del modelo iterativo e incremental del RUP, se construyen diversos prototipos, los cuales deberán pasar por un estricto control de calidad, definiendo con certeza cuales son los prototipos que cumplen con los requerimientos del usuario definido en etapas anteriores.

Existen muchos métodos de prueba de calidad de software, se recomienda el proceso de *ingeniería reversa.* Con este proceso probamos que la primera propuesta del diseño realizado en base al modelo de análisis es consecuente con la segunda propuesta del diseño, obtenido como producto del proceso de ingeniería reversa.

**2.1.7. ENTREGA**

Se gestiona el proceso de puesta en marcha del software, este debe estar preparado para la producción siendo flexible para facilitar el proceso de integración con otros sistemas de la organización.

Es requisito indispensable que los sistemas de información hayan aprobado todos los lineamientos de calidad, establecidos en la etapa de “***pruebas****”*, es suficiente la presencia de un error en el sistema de información para No entregar el software resultado.

**2.2. DISCIPLINAS DE SOPORTE**

**2.2.1. CONTROL DE CAMBIOS**

Establecimiento de políticas de gestión para la administración de cambios en el proyecto de construcción del software.

Los cambios generalmente vienen de los principales involucrados del proyecto “los clientes”, esos cambios son clasificados en 2 categorías:

• **Cambios relevantes**, aquellos que tienes repercusiones serias en el desarrollo del proyecto, incluso se puede modificar la estructura de la base de datos y la propuesta de interfaces.

• **Cambios irrelevantes**, aquellas que pueden ser solucionados sin mayor dificultad, este tipo de cambios no repercute en modificaciones mayores tanto en el ámbito de aplicación como en el ámbito de la base de datos.

Al inicio del proyecto se establecen las políticas de control de cambio, quién es la persona o personas integrantes del grupo de desarrollo que decepcionarán los pedidos de cambio; se resuelven las siguientes interrogantes ¿Cuáles son los formatos de recepción?, ¿Cuáles son los medios de recepción del cambio?, ¿Cuáles son las restricciones y supuestos, con respecto al manejo del cambio?

**2.2.2. GESTIÓN DE PROYECTOS**

El éxito de la construcción del software no solo depende del proceso, es necesario que el binomio administración del proyecto y proceso de construcción, funcionen de forma mancomunada, sólo así se logra el éxito en la construcción de software.

La gerencia de proyectos provee el marco que permite cumplir con los objetivos de la organización usando un proceso estructurado y controlado. Comprende varias técnicas, herramientas y metodología que permiten al gerente y su equipo llevar a cabo un proyecto que cumpla con el “principio del cuarto cuadrante4”.

El proyecto deberá cumplir:

• Con terminar en el tiempo pactado.

• Dentro de los límites de presupuesto.

• Con la calidad esperada por el cliente.

• Con el alcance establecido en la definición de proyecto.

El rol del gerente de proyectos es de gran responsabilidad, siendo el encargado de dirigir y supervisar el proyecto de principio a fin.

Algunas de sus principales tareas serán:

• **Definir el proyecto**: debe definir el alcance del proyecto, estableciendo sus límites, en otras palabras, se aclara que procesos, departamentos o elementos de la organización forma parte del proyecto.

Esto es fundamental para prevenir un crecimiento indeseado del proyecto, a medida que se progresa. Es importante diferenciar claramente aquellos elementos y resultados que son absolutamente necesarios, de aquellos que son deseables.

• **Planificar el proyecto**: planificar el proyecto implica proponer la solución a desarrollar, en base a los objetivos y resultados necesarios, y establecer cómo la desarrollará. Los puntos más importantes a considerar son: estrategia (cómo se relaciona el proyecto con el plan estratégico de la empresa), recursos (que necesito y con qué cuento), finanzas (cuánto costará y dónde obtener el dinero) y tiempo (de cuánto tiempo se dispone).

• **Obtener el respaldo de la alta gerencia**: para el éxito de cualquier proyecto, es fundamental el apoyo irrestricto de uno o más gerentes de alto nivel. Esto hará mucho más fluido todo el proceso, incluyendo la obtención de recursos, lograr la colaboración de toda la empresa y la resolución de conflictos entre departamentos si es posible.

• **Formar el equipo humano**: Identificar y ubicar a aquellas personas mejor calificadas para las distintas tareas involucradas. Con frecuencia, el equipo se forma con personas provenientes de distintas áreas de la organización, por lo que no reportan directamente al gerente del proyecto. En ocasiones, es necesario reforzar el equipo con personas de fuera del entorno de trabajo, en cuyo caso hay que hacer el reclutamiento.

• **Obtener los recursos**: Es responsabilidad del gerente de proyectos asegurar los recursos (dinero, equipos, personal de apoyo, espacio físico, etc.) que le permita al equipo funcionar en forma efectiva.

• **Definir las operaciones:** Incluye determinar las herramientas a utilizar (ej. software de manejo de proyectos), definir los canales de comunicación, establecer la logística, etc.

• **Controlar el proyecto**: Asegurar que las metas se están logrando y que el proyecto sigue el curso planificado.

En el transcurso del desarrollo del proyecto, surgirán cambios e imprevistos, en cuya circunstancia, es labor del gerente mantener la flexibilidad que le permita adaptarse, corregir y/o ajustar, sin poner en peligro los resultados.

Para evitar problemas de fracaso del proyecto, los gestores del proyecto deben prestar especial atención a las principales razones de fracaso:

• Mala definición o concepción del proyecto

• Cambios en el alcance o definición del proyecto

• Falta de una metodología adecuada para la administración del proyecto

• Falta de planificación en el control de los cambios

• Falta de comunicación entre los miembros del equipo entre ellos y el resto de la empresa

• Falta de claridad del contrato en términos de supuestos

y restricciones

• Desacuerdos entre clientes y los gerentes de proyecto

**2.2.3. ENTORNO**

El análisis del entorno, establece criterios o políticas que permitan el proceso de puesta en marcha del software construido. El nuevo software debe funcionar adecuadamente con los otros sistemas de información de la organización facilitando así el proceso de integración de los sistemas. El sistema de información necesita cumplir con la factibilidad técnica5 y operativa6, para lograr el éxito en la organización.

**3. Organización y elementos en RUP**

Ya conociendo varias partes del RUP nos concentraremos ahora en los elementos que lo componen, entre estos se tienen: Flujos de Trabajo, Detalle de los Flujos de Trabajo, Actores, Actividades y Artefactos. En la figura 1 se muestran más claramente cómo se representan gráficamente cada uno de estos elementos y la interrelación entre ellos. Se puede observar que el Flujo de Trabajo de Requerimientos conlleva varios pasos, cada uno de estos pasos tiene asociado uno o varios actores, los cuales a su vez son los encargados de la ejecución de varias actividades, las cuales a la vez están definidas en artefactos o guías para su realización.

**3.1.** **Actores o roles**

Son los personajes encargados de la realización de las actividades definidas dentro de los flujos de trabajo de cada una de las disciplinas del RUP, estos actores se dividen en varias categorías: Analistas, Desarrolladores, Probadores, Encargados, Otros actores.

Los actores NO son los individuos, los actores describen cómo los individuos deben comportarse en el negocio y qué responsabilidades deben tener.

El RUP establece 30 roles diferentes. Una de las preguntas más comunes es ¿necesitamos 30 personas como mínimo para abordar la construcción de un software?, la respuesta obviamente es NO, una persona puede realizar más de un rol en el desarrollo del proyecto, eso dependerá de su experiencia, conocimiento y habilidades; se debe tener especial cuidado en la asignación de roles, este trabajo debe ser realizado por el ingeniero de software con colaboración y comunión del equipo de desarrollo, una persona no puede asumir dos roles dependientes, es decir una misma persona NO puede ser juez y parte.

A continuación se presenta una lista de actores de acorde a las categorías mencionadas con anterioridad:

**Analistas**

* Analista del Proceso del Negocio
* Diseñador del Negocio
* Revisor del Modelo del Negocio
* Revisor de Requerimientos
* Analista del Sistema
* Especificador de Casos de Uso
* Diseñador de Interfaz del Usuario

**Desarrolladores**

* Arquitecto
* Revisor de la Arquitectura
* Diseñador de Cápsulas
* Revisor del Código y Revisor del Diseño
* Diseñador de la Base de Datos
* Diseñador
* Implementador y un Integrador

**Probadores Profesionales**

* Diseñador de Pruebas
* Probador

**Encargados**

* Encargado de Control del Cambio
* Encargado de la Configuración
* Encargado del Despliegue
* Ingeniero de Procesos
* Encargado de Proyecto
* Revisor de Proyecto

**Otros**

* Cualquier trabajador
* Artista Gráfico
* Stakeholder
* Administrador del Sistema
* Escritor técnico
* Especialista de Herramientas

**3.2 ARTEFACTO:**

Definido como la pieza de información que es producida, modificada, o utilizada por un proceso en particular, son productos tangibles del proyecto, usados por los trabajadores para realizar nuevas actividades y son el resultado de esas actividades. Pueden ser los siguientes:

• El documento, donde se especifiquen a los casos de uso de negocio o donde se define la arquitectura del software.

• El modelo, como el modelo de caso de uso, modelo de análisis, modelo de diseño, etc.

• Un elemento dentro de un modelo tal como una clase o un sub sistema.

Ahora se mencionan cuáles son los artefactos a utilizar en cada de una de etapas de construcción de software según el

RUP, tanto en las etapas centrales como de soporte.

No se trata de memorizar que artefacto producirá un trabajador, el uso constante de estos hará que sean parte tácita del desarrollo de software, sólo tenemos que aplicarlo.

**3.2.1 Conjuntos de artefactos**

Se tiene un conjunto de artefactos definidos en cada una de las disciplinas y utilizadas dentro de ellas por lo actores para la realización de las mismas, a continuación se definen cada una de estas categorías o grupos de artefactos dentro de las disciplinas del RUP:

1. Modelado del negocio

Los artefactos del modelado del negocio capturan y presentan el contexto del negocio del sistema. Los artefactos del modelado del negocio sirven como entrada y como referencia para los requisitos del sistema.

b) Requerimientos

Los artefactos de requerimientos del sistema capturan y presentan la información usada en definir las capacidades requeridas del sistema.

c) Análisis y diseño del sistema

Los artefactos para el análisis y del diseño capturan y presenta la información relacionada con la solución a los problemas que se presentaron en los requisitos fijados.

d) Implementación

Los artefactos para la implementación capturan y presentan la realización de la solución presentada en el análisis y diseño del sistema.

e) Pruebas

Los artefactos desarrollados como productos de las actividades de prueba y de la evaluación son agrupadas por el actor responsable, con el cual se lleva un conjunto de documentos de información sobre las pruebas realizadas y las metodologías de pruebas aplicadas.

f) Despliegue

Los artefactos del despliegue capturan y presentan la información relacionada con la transitividad del sistema, presentada en la implementación en el ambiente de la producción.

g) Administración del proyecto

El conjunto de artefactos de la administración del proyecto capturan los artefactos asociados con el proyecto, el planeamiento y a la ejecución del proceso.

h) Administración de cambios y configuración

Los artefactos de la configuración y administración del cambio capturan y presentan la información relacionada con la disciplina de configuración y administración del cambio.

1. Entorno o ambiente

El conjunto de artefactos del ambiente presentan los artefactos que se utilizan como dirección a través del desarrollo del sistema para asegurar la consistencia de todos los artefactos producidos.