

1. Buenas Prácticas de RUP

(extraído de A Manager's intro to RUP, se recomienda leer los enlaces adicionales que da este documento)

a) Adaptar el proceso:

- RUP es una metodología, compuesta de diversos tipos de documentación; ésta se debe de adaptar al sistema y al tipo de negocio analizado. No hay que seguir toda la documentación en todos los casos.
- Ejemplo: Ingrese al siguiente enlace:
<http://sce.uhcl.edu/helm/rationalunifiedprocess/>
Luego seleccione Disciplines/Business Modeling, luego haga click en el botón Concepts e ingrese a Scope of Business Modeling.

b) Balancear las prioridades de los stakeholders:

- Se debe de mantener un contacto con los stakeholders, de tal manera que las necesidades del proyecto a tener sean prioritarias. Se pueden hacer negociaciones entre las necesidades de usuario de estos integrantes.

c) Colaboración entre equipos:

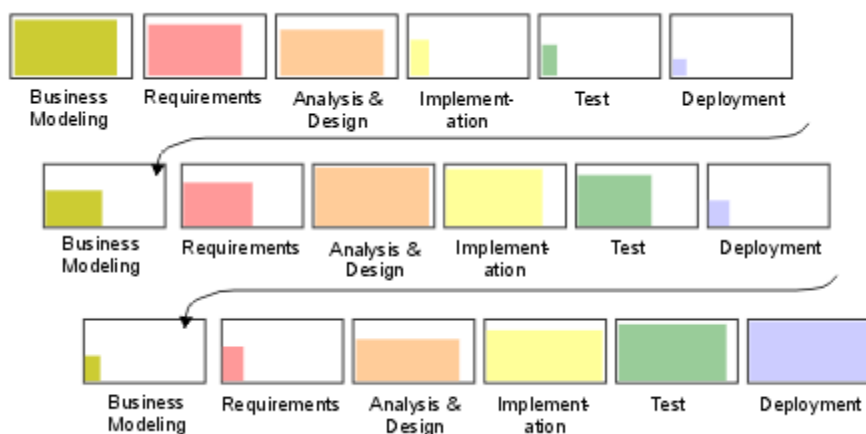
- Debe de existir una comunicación efectiva entre los stakeholders y los desarrolladores. Puede darse que estén integrados.

d) Demostrar los avances de forma iterativa

- Esto permite que los stakeholders tenga una visión de cómo se va desarrollando el sistema, a la vez que se minimiza el riesgo de retrasos debido a características no realizadas en el sistema. Si desea conocer un poco más sobre el concepto de Iteraciones ingrese al enlace anteriormente dado de: <http://sce.uhcl.edu/helm/rationalunifiedprocess/> y seleccione la parte de Iterations.

Pregunta:

Interprete las siguiente gráfica:



e) Elevar el nivel de abstracción:

- Utilizar herramientas de modelado y reutilizar componentes.

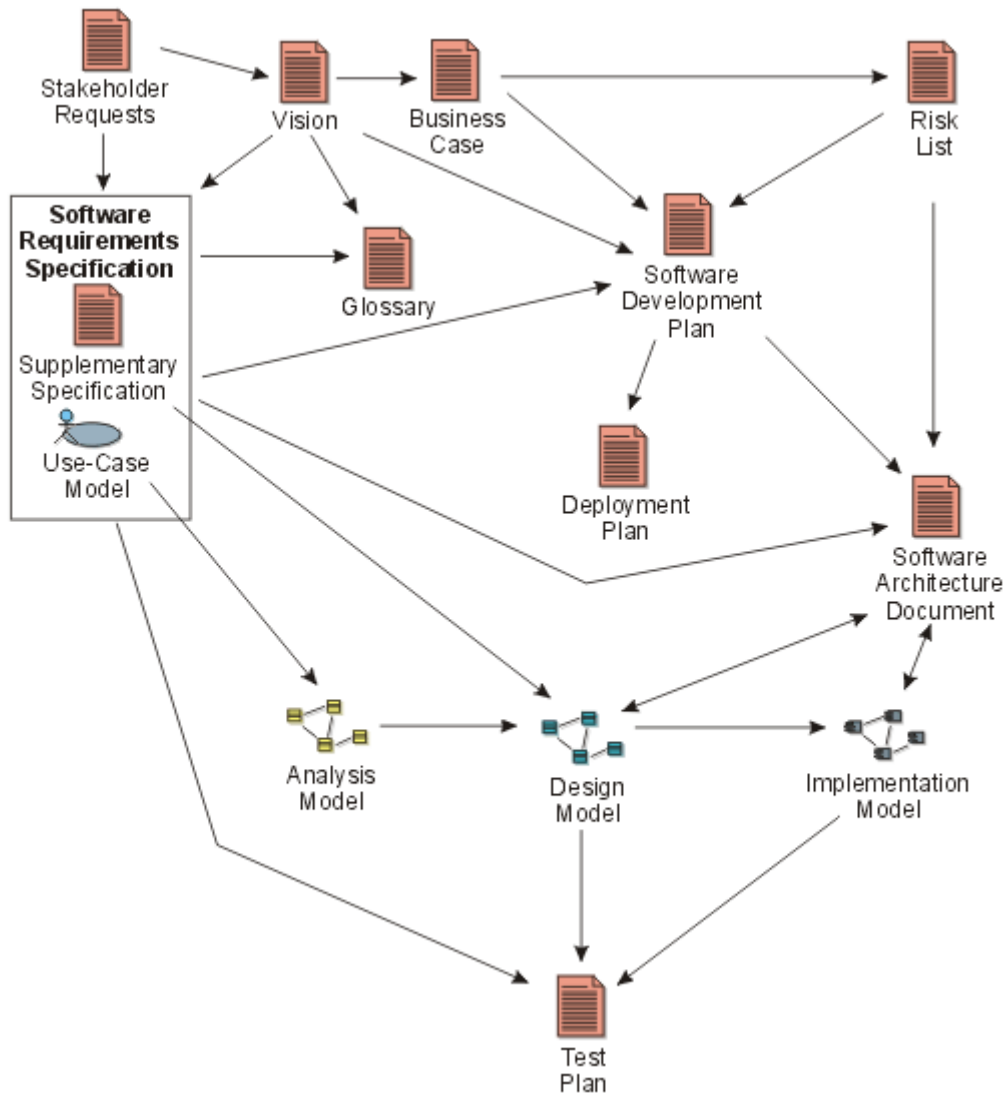
f) Enfocarse en la calidad:

- Esto debe de darse en todas las etapas del sistema, no sólo es responsabilidad de los equipos de prueba.

- Existen diversas técnicas pudiendo ser una de ellas el Test Driven Development.

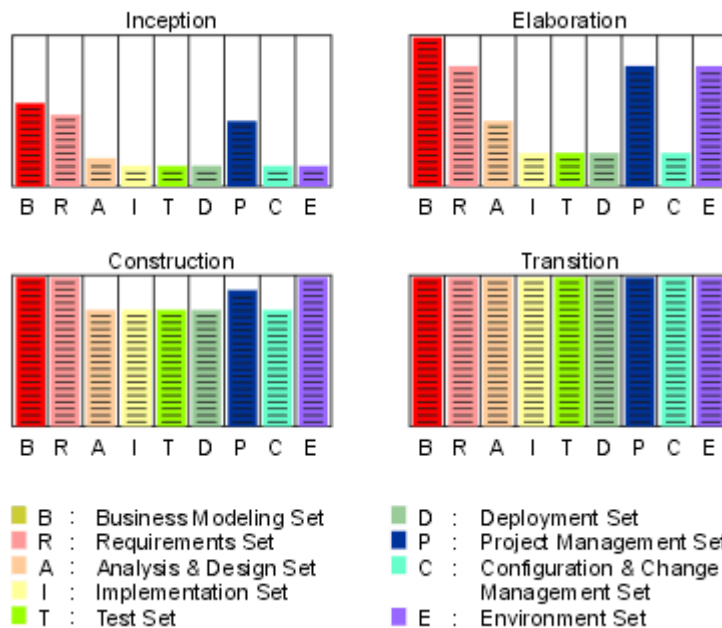
2. Artefactos en RUP

Según http://sce.uhcl.edu/helm/rationalunifiedprocess/process/artifact/ovu_arts.htm, los artefactos son entregables que se producen en diversas etapas de un proyecto. Un esquema general estaría dado por:



Pregunta:

Ingresando al enlace de Iterations interprete la siguiente gráfica:



y diga porqué se manifiesta que el conjunto de artefactos, set of artifacts, se vuelve más maduro conforme avanzan las iteraciones.

Pregunta: Revise los artefactos que se requieren para los modelos, cómo se llaman éstos? cuál es el esquema que se solicita para un modelo de casos de uso?


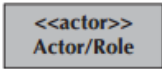

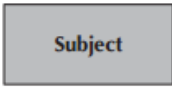

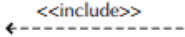
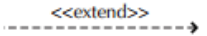

Pregunta: De los patrones de iteración, cuál se adecuaría más al proyecto que usted ha escogido?

Adicionales de apoyo para su proyecto:

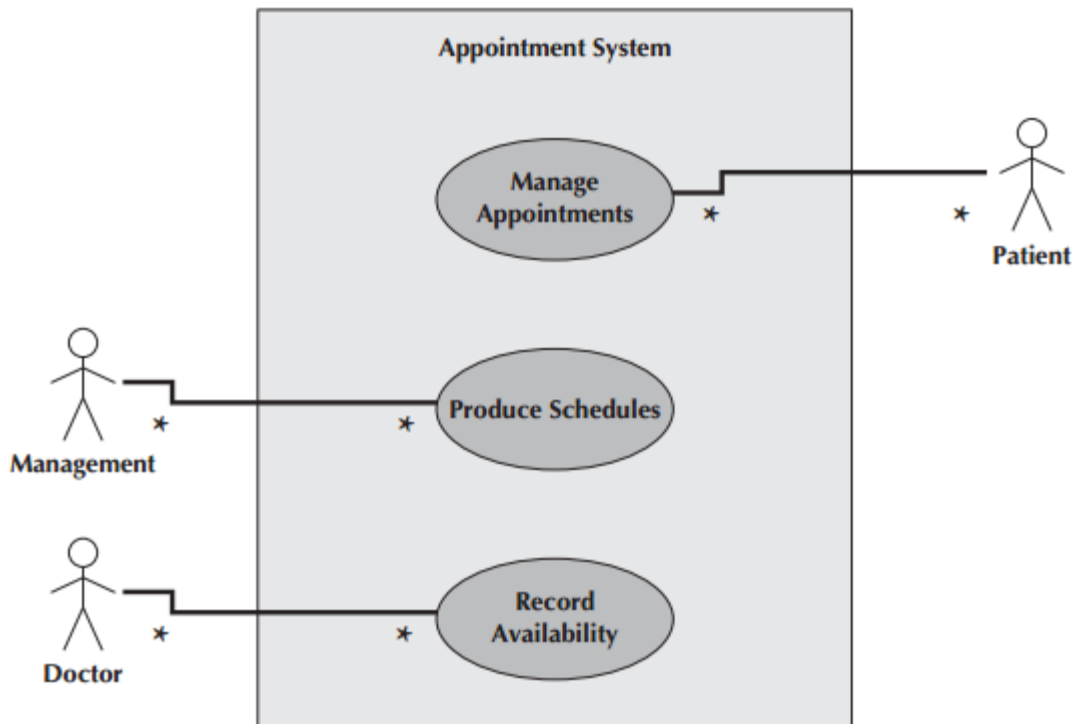
- Se han colocado plantillas para el modelo Canvas solicitado (canvas esp, canvas temp)
- Consejos respecto a su proyecto se ha colocado en el archivo llamado adic_proyecto
- Para la parte del modelado del negocio AS-IS puede usar algunas de las formas, a excepción de BPM que será usado para otra parte de su proyecto, del siguiente enlace:
<http://creately.com/blog/diagrams/business-process-modeling-techniques/>
- Para la parte de definición del modelo AS-IS de una manera más detallada y su relación con BPM puede leer el archivo denominado AS_IS_BPM. Esta sección se ha extraído del libro: "The Complete Business Process Handbook", de Mark von Rosing.

Casos de Uso:

Forma de representación:

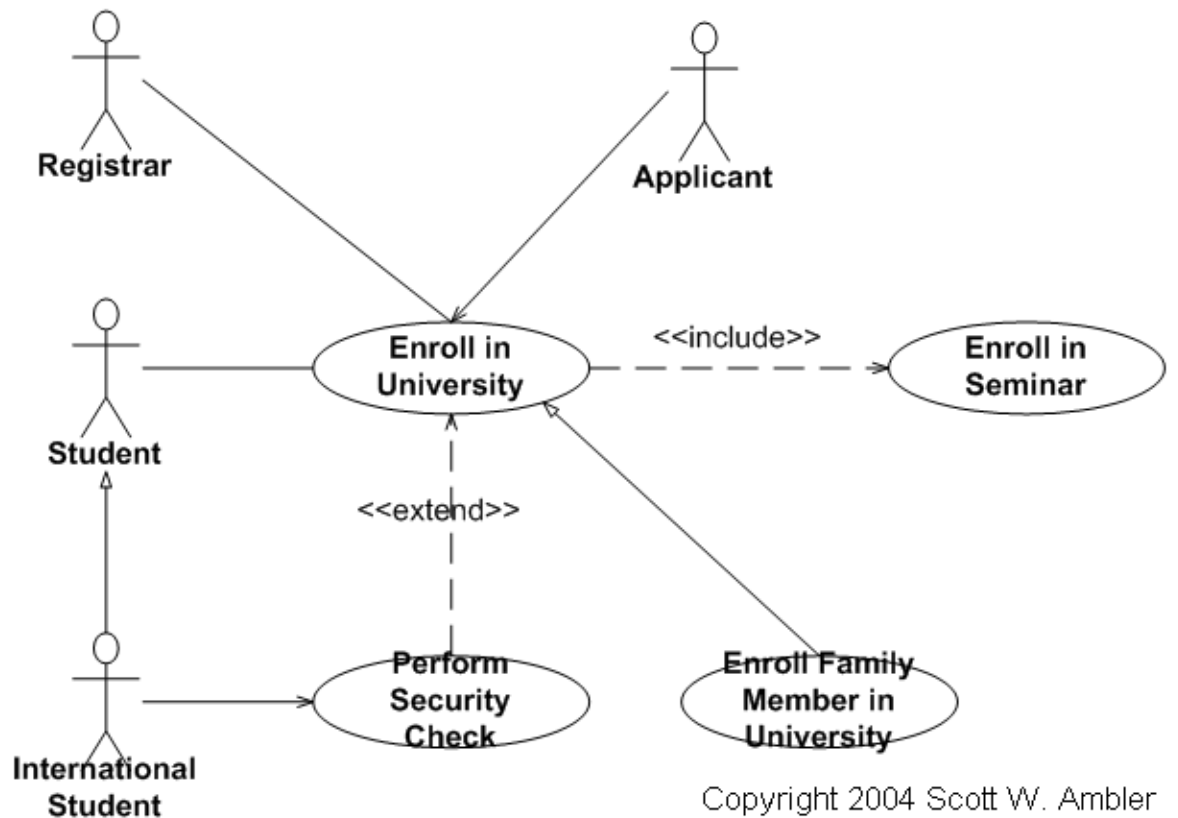
<p>An actor:</p> <ul style="list-style-type: none"> ■ Is a person or system that derives benefit from and is external to the subject. ■ Is depicted as either a stick figure (default) or, if a nonhuman actor is involved, a rectangle with <<actor>> in it (alternative). ■ Is labeled with its role. ■ Can be associated with other actors using a specialization/superclass association, denoted by an arrow with a hollow arrowhead. ■ Is placed outside the subject boundary. 	 <p>Actor/Role</p> 
<p>A use case:</p> <ul style="list-style-type: none"> ■ Represents a major piece of system functionality. ■ Can extend another use case. ■ Can include another use case. ■ Is placed inside the system boundary. ■ Is labeled with a descriptive verb-noun phrase. 	
<p>A subject boundary:</p> <ul style="list-style-type: none"> ■ Includes the name of the subject inside or on top. ■ Represents the scope of the subject, e.g., a system or an individual business process. 	
<p>An association relationship:</p> <ul style="list-style-type: none"> ■ Links an actor with the use case(s) with which it interacts. 	
<p>An include relationship:</p> <ul style="list-style-type: none"> ■ Represents the inclusion of the functionality of one use case within another. ■ Has an arrow drawn from the base use case to the used use case. 	
<p>An extend relationship:</p> <ul style="list-style-type: none"> ■ Represents the extension of the use case to include optional behavior. ■ Has an arrow drawn from the extension use case to the base use case. 	
<p>A generalization relationship:</p> <ul style="list-style-type: none"> ■ Represents a specialized use case to a more generalized one. ■ Has an arrow drawn from the specialized use case to the base use case. 	

Ejemplo:



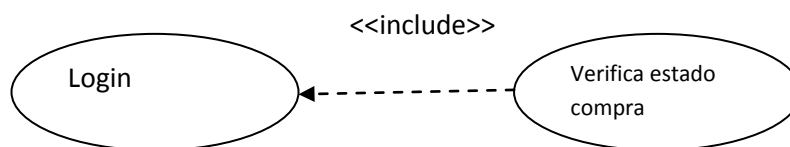
Apuntes adicionales sobre Casos de Uso:

- Diferencia entre extends e include (extraído de: <http://agilemodeling.com/essays/useCaseReuse.htm>)
- Extend, es un tipo de generalización y se da cuando continúa el comportamiento del caso de uso base. Después de terminar su ejecución debería devolverse el mando al caso de uso base.
- Include, es igualmente una forma de generalización, pero en este caso el comportamiento de un caso de uso posee bastantes similitudes con otro caso de uso. Mayor información sobre ambos casos se encuentra en: <http://www.uml-diagrams.org/use-case-include.html>



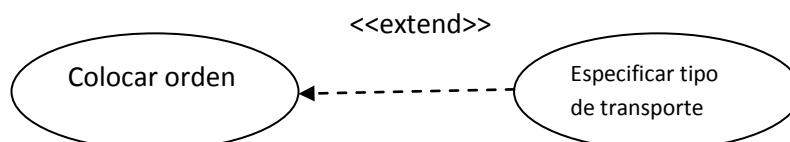
Otros ejemplos:

Se desarrolla un sistema de compras en línea, para esto un usuario puede *hacer login al sistema*, cuando hace su compra también puede *revisar el estado de la compra*:



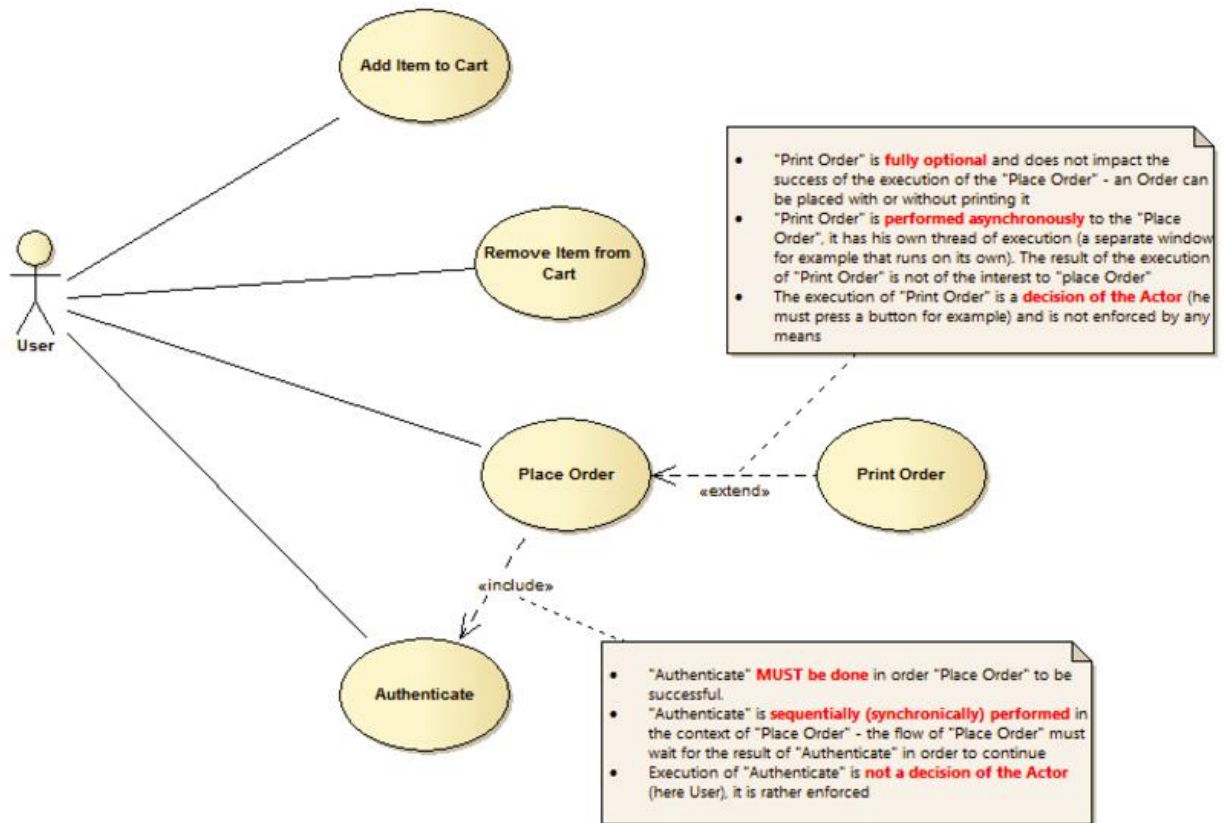
- Para verificar el estado de la comprar siempre tiene que haber un login primero
- El caso de uso de login puede existir para otros casos de uso.

En el mismo sistema de compras en línea cuando una persona *coloca su orden* puede *especificar el tipo de transporte*:



- Se podría decir que especificar el tipo de transporte es opcional.
- Permite agregar funcionalidad cuando se dan ciertas restricciones.

Otro ejemplo genérico:



Resumen de extend e include:

- Include se utiliza cuando hay partes de un sistema que son comunes.
- El caso base es dependiente del caso que incluye, por ejemplo:



- El caso base estaría incompleto sin el caso base.
- Extend es para casos adicionales.
- El caso base extendido no es dependiente de la extensión.
- El caso base se encuentra completo sin la extensión del sistema.

Ejercicios:

Puede modelar sus casos de uso utilizando la siguiente herramienta online:

<https://www.glimpy.com/go/html5/launch>

1. Un hospital desea diseñar un sistema de citas de pacientes. Un paciente puede hacer su cita mediante un programa en línea que permite separar su cita con un médico cualquiera. Un administrador es el encargado de realizar los horarios de atención de los médicos. Para la atención de los pacientes los doctores deberán ser capaces de registrar su disponibilidad de forma semanal mediante este sistema. Modelar los actores y casos de uso para este sistema de citas.

2. Se desea hacer un sistema para el manejo de citas en un hospital. Los pacientes que realizan sus citas pueden ser catalogados como pacientes nuevos o antiguos; para el caso de los pacientes nuevos se les deberá crear una nueva ficha o historia clínica y puede darse que un paciente antiguo se tenga que modificar sus datos personales. Las formas de pago son variadas y se le da al paciente un conjunto de opciones a fin de poder seleccionar la que mejor le convenga.

Modelar el sistema descrito colocando los actores, casos de uso y relaciones de extend e include según correspondan. Puede tomar como base el modelo de casos de uso de la pregunta 1.

Descripción de Casos de Uso

(ver adjunto)

Referencias

Dennis, A., Wixom, B. H., Tegarden, D. P., & Seeman, E. (2015). System analysis & design: an object-oriented approach with UML. Hoboken, NJ: Wiley.