CC51A – Ingeniería de Software

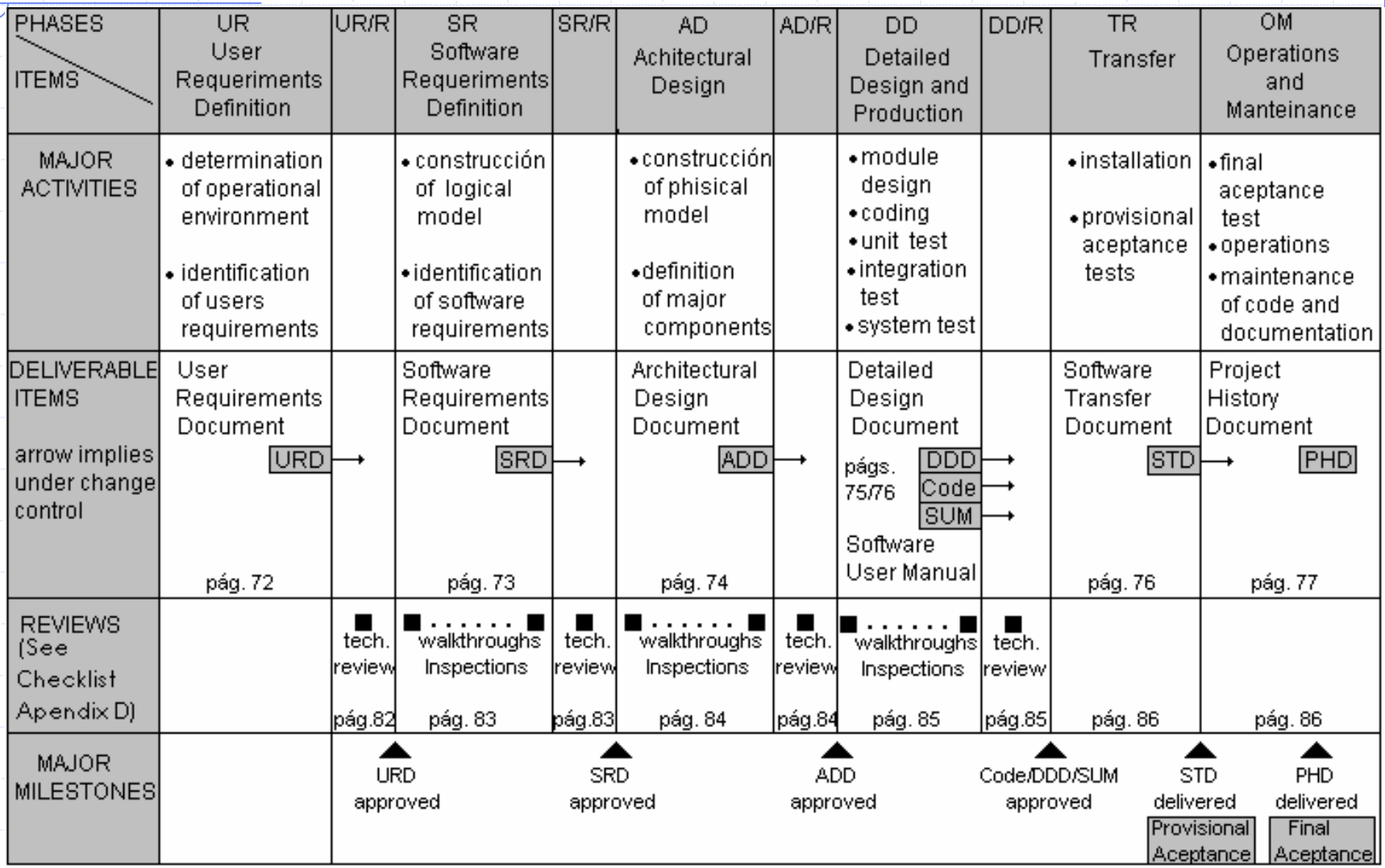
Especificación de Requisitos de Usuario y de Software

Sergio Ochoa D.

Estructura de la Presentación

- Definición de Requisitos de Usuarios.
- Definición de Requisitos de Software.
- Administración de Requisitos durante un Proyecto.
- Bibliografía y casos.

Ciclo de Vida ESA



Fase de Definición de Requisitos de Usuarios

- El propósito es refinar una idea sobre una tarea a realizarse, utilizando equipo computacional, hacia una definición de lo que se espera del sistema computacional.
- Los analistas son los responsables de esta tarea.
- Para identificar estos requisitos, hay que entrevistar al cliente/usuario.
- Debe usarse la experiencia de ingenieros, personal clave (cliente/usuario), y especialistas para ayudar a especificar/revisar los requisitos.

Fase de Definición de Requisitos de Usuarios

- Su entregable es el URD, el cual es crítico para el resto del proyecto porque define la base sobre la cual se aceptará el software.
- Esta fase termina con aprobación formal del URD por la actividad de revisión UR/R.
- El URD representa la visión del usuario, acerca del problema a resolver.

UR: Entradas a la Fase

- No hay entradas formales para esta fase.
- … aunque es posible utilizar informes, proyectos, documentos o sistemas de software anteriores, para iniciar esta fase.

UR: Actividades de la Fase

- Capturar y Especificar los requisitos del usuario.
- Determinar el Ambiente Operacional.
- Revisar el URD (en hitos preestablecidos), hasta lograr su aprobación.

Capturar los Requisitos de Usuarios

- Proceso iterativo.
- Se realizan entrevistas con el cliente/usuario.
- Se usan cuestionarios.
- Se revisan todos los formularios y sistemas involucrados.
- Se establecen acuerdos con respecto a los límites del sistema.

Determinación del Ambiente Operacional

- Primer paso para definir los requisitos de software.
- Debe establecerse el escenario en el que operará el software a desarrollar.
- Es una narrativa que puede apoyarse de diagr. de contexto o de bloques, que permita mostrar el uso del sistema en un ambiente más grande.
- Cosas a revisar en la def. del escenario: tipos de máquinas (hard y soft), tipos de comunic. a usar (telef., LAN, WAN, etc.), interfaces con otros sist. (en producc. o en desarrollo) involucrados en el escenario de trabajo.

Especificación de Requisitos de Usuarios

Los tipos de requisitos de usuarios son:

- ***de Capacidades*** requeridas por los usuarios para resolver un problema o determinar un objetivo.
- ***de Restricciones*** impuestas por los usuarios, sobre la forma como debe ser resuelto el problema, o logrado el objetivo.

Especificación de Requisitos de Usuarios

Algunos problemas en la definición de URs son:

- Correctitud.
- Completitud.
- Consistencia.
- Independencia de requisitos.
- Ambigüedad de los requisitos.

Ejemplo de Especificac. de Req.

NOMBRE: Consistencia

ESCALA: probabilidad de que dos mediciones básicas sean consistentes

PRUEBA: 1000 mediciones básicas comparadas usando 3 cifras significativas

ACTUAL: entre 90% y 95%

PEOR: 98%

PLANIFICADO: 99.5%

AUTORIDAD: minuta del 20/12/2005

Otro Ejemplo de Especificación de Requisitos

NOMBRE: Consistencia

ESCALA: probabilidad de que se muestre un dato inconsistente, sin que se advierta su inconsistencia.

PRUEBA: sobre una base de datos 100% consistente, se ingresan inconsistencias hasta hacerla 98% inconsistente. Luego se cuentan los datos inconsistentes no detectados.

ACTUAL: 100% (no hay detección).

PEOR: 5%.

PLANIFICADO: 1%.

AUTORIDAD: Entrevista con Usuario (05/12/2005).

La Escala

La escala: es la medición que se hará de un atributo.

- Debe decir que es lo que se mide, y en que unidad de medida.
- Es una medición teórica (por ej. Kilogramos).
- La medición práctica asociada se llama *prueba* (por ej. uso de una pesa).
- La escala dice *qué?* se mide.
- La prueba dice cómo? se mide.

Veamos el siguiente ejemplo de **Escala...**

La Escala (cont...)

Supongamos el atributo **Amistosidad:**

- Facilidad de Aprendizaje
  - ESCALA = minutos de aprendizaje
  - PLAN = 5
- Productividad
  - ESCALA = tareas por hora
  - PLAN = 20

La **amistosidad** no tiene escala, se define a través de sus subatributos: facilidad de aprendizaje, y productividad.

La Escala (cont...)

- Hay que definir las cosas en aquel nivel de detalle que permita controlar los parámetros críticos del sistema.
- Los conceptos abstractos como “amistosidad”, **normalmente** necesitan explosión.
- Conceptos ambigüos como “mejor que el sistema actual”, **siempre** necesitan explosión.

La Prueba

La prueba: es el modo de medir un atributo.

- Idealmente la prueba se refiere a conceptos ya conocidos medidos en la organización.
- Es mejor partir con una prueba poco precisa, antes que partir sin una prueba.
- Después se podrá mejorar….
- La pruebas tienen que especificarse por escrito.

“ más vale una prueba mediocre por escrito, que una prueba perfecta sin especificar (que está en la cabeza de alguien)”

La Prueba (cont...)

- Se reconoce el costo de la medición (para el presupuesto y el plan).
- Todos sabrán que el alcance del objetivo se puede y se va a medir.
- Se puede preparar un punteo preliminar de un plan de pruebas para los atributos
  - Si no se dice lo contrario, la prueba es aplicable durante las pruebas de aceptación
  - Los tiempos pueden especificarse entre paréntesis. Por ej.

    PRUEBA (diseño) = usar inspecciones

    PRUEBA (codificación) = usar pruebas de unidades

    PRUEBA (producción) = usar datos de uso del sistema

El Peor Caso

El peor caso: es el peor nivel aceptable bajo cualquier circunstancia.

- Cualquier valor menor significa que hubo fracaso
- Obviamente, para tener éxito no podemos tener todos los atributos en este nivel.
- La idea es llegar al nivel planificado

El **peor caso** de cada atributo debe ponerse en un contrato legal.

- Si no se cumple, el proveedor está en deuda
- Se podrían incluso definir multas
- Todas las partes interesadas deben firmar: cliente, proveedor, usuarios, etc.

El Peor Caso (cont…)

Para determinar el peor caso:

- Imagine un nivel que sea siempre inaceptable.
- Empiece a subir el nivel hasta llegar a un punto en donde en “ciertos casos” sería aceptable.
- Si es necesario defina más de un peor caso. Uno para cada período. Por ejemplo:

  PEOR (marcha blanca) = 80%

  PEOR (operación inicial) = 95%

  PEOR (después de garantía) = 99%

El Nivel Planificado

El nivel planificado: es aquel nivel que se desea alcanzar, en conjunto con los demás atributos.

- Este es el límite de la definición de éxito.
- Una vez que este límite es alcanzado, ya no es necesario asignar más recursos allí.
- Alcanzar el nivel planificado no significa “perfección”.
- Éste es un nivel que debe ser alcanzable y realista.
- Debe ser consistente con los niveles planificados de los demás atributos.

El Nivel Planificado (cont…)

¿Cómo se define el nivel planificado?

- No es fácil, y es un proceso iterativo
- Es muy similar al proceso de hacer un presupuesto, o una carta Gantt.
  - El nivel planificado del recurso “plata”, es el presupuesto.
  - El nivel planificado del recurso “tiempo” es la duración estimada del proyecto.
- Ideas que ayudan:
  - Usar información de proyectos similares
  - Usar información histórica de la empresa
  - Usar información de productos de la competencia.

El Nivel Planificado (cont…)

¿Cómo se define el nivel planificado?

- Siempre conviene partir con niveles planificados definidos, aunque sea a ojo.
- En general es sano ser un “poco” optimistas.
  - Así obligamos a los diseñadores a pensar más
  - Eso nos permite mejorar nuestros productos
  - Hay que estar dispuesto a retractarse
- Los usuarios/clientes/desarrolladores deben entender que son estimaciones.

ADVERTENCIA: Los niveles especificados en este documento representan las mejores estimaciones disponibles hasta la fase de ***. No comprometen a ninguna de las partes involucradas, excepto que se diga explícitamente lo contrario. Sólo existen para ayudar a encontrar soluciones factibles.

El Nivel Planificado (cont…)

- Lo que hace difícil pensar en los niveles planificados es que deben satisfacerse todos.
- Para establecer el nivel planificado de un atributo, hay que considerar todos los atributos que interactúan con él.
  - Es aquí donde se consideran las interacciones
  - Es aquí donde se evalúan las opciones del cliente
  - Es aquí donde se considera la factibilidad
  - En el peor de los casos, basta ver cada atributo por separado

“ Si un sistema no necesita ser confiable, puede satisfacer cualquier otro requisito”

Ley cero de la confiabilidad de Weinberg.

El Mejor Caso

El mejor caso: es el mejor valor que se sabe ha sido alcanzado en alguna situación.

- También se llama nivel *record*.

- Para calificar se usan paréntesis

  Ejemplo:

  - MEJOR (competidor xx) = 80% (revista YY, Nov.2000)
  - MEJOR (competidor xx, futuro) = 90% (revista YY, Nov.2000)
  - MEJOR (prototipo ideal, nuestro) = 95%

- El uso del mejor caso es opcional
  - Sirve para informar los límites de la ingeniería actual
  - No es una idea irrealizable (operación 100% continua).

El Nivel Actual

El nivel actual: es el nivel de un atributo contra el que queremos comparar.

- Ayuda a entender, evaluar, y aprobar los otros niveles.
- Aún cuando no hay un sistema que se reemplace, el proceso o su equivalente, se está haciendo. De ahí se puede tomar el nivel actual.
- La comparación también puede hacerse con la competencia u otros sistemas.

- El nivel actual es opcional.

El Nivel Actual (cont…)

Diferencia entre “nivel actual” y “mejor nivel”:

- El nivel “actual” es desde donde partimos.
- El “mejor” nivel no es el punto de partida, sino el nivel más alto que alguna vez se ha medido.
- Usualmente, el nivel “planificado” es mejor que el “actual”.
- Casi siempre el nivel “planificado” es peor que el “mejor” nivel.
  - La excepción es cuando se quiere hacer avanzar el estado del arte.

La Autoridad

La autoridad: es una indicación que sustenta el requisito.

- Puede ser la orden de un jefe
- Puede ser un manual de procedimientos
- Puede ser un requisito ya aprobado.
- Puede ser un contrato
- O incluso, una ocurrencia basada en la experiencias previas.

- La autoridad es opcional y puede haber más de una. Por ejemplo:

  AUTORIDAD = Documento de análisis de Juan Pérez (debe ser revisado!!)

  AUTORIDAD = Anexo A del contrato, sección 5.6b.

Ejemplos de Requisitos de Calidad

- Estos atributos deberían ser revisados a la hora de especificar un sistema. Debido a que aparecen en la mayoría de ellos.

- Entre los *Requisitos de Calidad Típicos* figuran:

Funcionalidad:	Usabilidad:
Pertinencia	Entendibilidad
Precisión	Aprendibilidad
Interoperabilidad	Operabilidad
Adherencia	Aceptación de Uso
Seguridad	

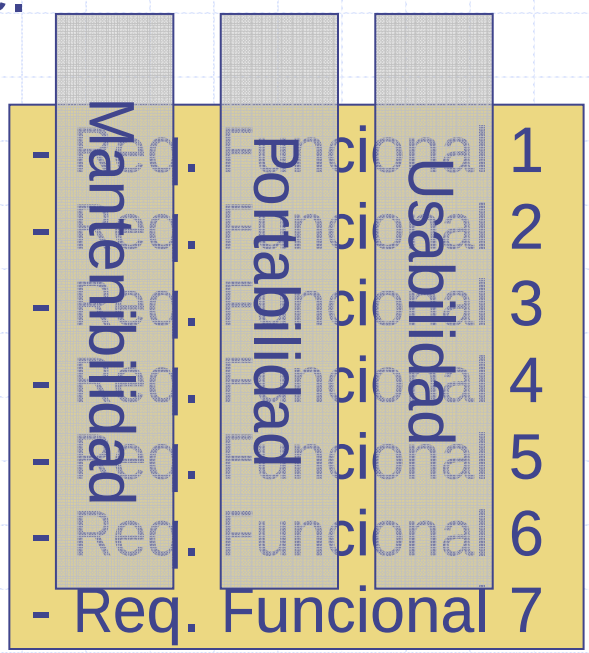
Ejemplos de Requisitos de Calidad (cont...)

- Entre los *Requisitos de Calidad Típicos* figuran:

Mantenibilidad: Analizabilidad Cambiabilidad Estabilidad Demostrabilidad	Portabilidad: Adaptabilidad Instanciabilidad Adecuación Reemplazabilidad
Eficiencia: Rendimiento Uso de Recursos	Confiabilidad: Madurez Tolerancia a Fallas Recuperabilidad

Requisitos de Calidad (cont…)

- Los requisitos de calidad son transversales a los requisitos funcionales.
- También se los suele llamar “aspectos de diseño” de un software.



Especificación de Requisitos de Usuarios

Atributos de requisitos de usuarios:

- La forma completa para especificar requisitos es la que vimos en las clase.
- Por razones de background del equipo de trabajo, y de tiempo para desarrollar el proyecto, especificaremos los requisitos de la siguiente manera:
- Para Requisitos Funcionales y de Restricción, especificar:
  - ***Identificador***. Identifica unívocamente al requisito.
  - ***Descripción.*** Describe el requisito.
  - ***Prioridad.*** Indica una prioridad. Por ej.(alta-media-baja)
  - ***Fuente.*** Referencia a la fuente de donde surgió el requisito (documento, grupo, entrevista, etc.).

Especificación de Requisitos de Usuarios

Atributos de requisitos de software:

- Para Requisitos de Calidad relevantes, utilizar el formato completo.

- Para Requisitos de Calidad poco relevantes, utilizar el mismo formato que para los Requisitos Funcionales y de Restricción.

Revisiones del URD

- Las salidas de esta etapa deben ser formalmente revisados durante la actividad de revisión de Requisitos de Usuarios (UR/R).
- Están involucrados los: usuarios, clientes, analistas, administ. del proyecto, y los ténsters.
- Es aconsejable realizar revisiones periódicas, antes de la revisión final.
- Eso mejorará la calidad, y la disponibilidad del producto final de la fase (URD).

UR: Salidas de la Fase

Documento de Requisitos de Usuario (URD).

1. Introducción

1.1 **Propósito.** Narrativa que describe el propósito del sistema a desarrollar.

1.2 **Alcance.** Narrativa que establece el alcance del sistema a desarrollar.

1.3 **Definiciones, acrónimos y abreviaciones.** Listado de acrónimos, con su significado.

1.4 **Referencias.** Se colocan las referencias de los documentos utilizados para construir el URD.

1.5 **Descripción general.** Narrativa que describe a grandes rasgos, el sistema a desarrollar.

UR: Salidas de la Fase

Documento de requisitos de usuarios.

2. Descripción General

2.1 Perspectiva del producto. Narrativa que establece que es lo que se espera obtener con el producto de software en desarrollo.

2.2 Características de los usuarios. Narrativa que establece los diferentes tipos de usuarios que interactuarán con el software, y las características de cada uno de ellos (atribuciones, y limitaciones).

2.3 Restricciones generales. Narrativa que establece cuales son las restricciones de funcionamiento del software. Por ej. que funcione en Windows y UNIX.

UR: Salidas de la Fase

Documento de requisitos de usuarios.

2. Descripción General

2.4 Suposiciones y dependencias. Narrativa que establece (por escrito) todas aquellas suposiciones, que se han hecho (con el consentimiento del cliente/usuario), sobre alguna característica del software a desarrollar.

2.5 Ambiente operacional. Narrativa que establece las características del escenario de trabajo del software.

UR: Salidas de la Fase

Documento de requisitos de usuarios.

3. Requisitos Específicos

3.1 Requisitos de Capacidad. Listado de requisitos que especifican las capacidades que debe poseer el software (lo que debe poder realizar el software), para satisfacer las necesidades del cliente/usuario. Estos requisitos se deben especificar según el formato preestablecido.

3.2 Requisitos de Restricciones. Listado de requisitos que especifican restriciones a cerca de cómo debe ser construido (se refiere a conexiones con otros sistemas, o adhesión a los estándares de la empresa) y operado el software. Estos requisitos se deben especificar según el formato preestablecido.

Definición de Requisitos de Software ....

Definición de Requisitos de Software (SR).

- Fase de análisis del problema.
- El propósito es analizar las definiciones de requisitos de usuarios en el URD, y producir un conjunto de requisitos de software lo más completo, consistentes y correctos posibles.
- El SRD contiene una visión del problema, desde el punto de vista del desarrollador, y no del usuario.
- Pueden usarse prototipos para validar los requisitos (es recomendable).

SR: Entradas a la Fase

- Documento de Requisitos de Usuarios (URD);
- El estándar establece otras entradas, pero no las consideraremos para nuestro desarrollo.

SR: Actividades de la Fase

- Construcción del Modelo Lógico.
- Especificación de los Requisitos de Software (SRD).
- Revisiones del SRD.

SR: Actividades de la Fase

Construcción del Modelo Lógico:

- Construcción de un modelo independiente de la implementación del sistema requerido por el usuario.
- Puede ser construído por descomposición “top-down” de la función principal, inferido del URD, en una jerarquía de funciones.
- Caminatas (“Walkthroughs”), revisiones e inspecciones para asegurar que se cumplen las especificaciones de cada nivel antes de descender al siguiente.

Actividades de la Fase

Construcción del modelo lógico debe satisfacer las siguientes reglas:

- Las funciones deben tener un propósito único. Sus nombres deben tener una estructura declarativa (qué hacen en vez de cómo lo hacen).
- Las funciones deben estar en el nivel apropiado en el que aparecen.
- Las interfaces entre módulos deben minimizarse.

Actividades de la Fase

Construcción del modelo lógico debe satisfacer las siguientes reglas:

- Cada función debe descomponerse en no más de 7 sub-funciones.
- El modelo debe omitir información de implementación.
- Deben especificarse los atributos de rendimiento de cada función (capacidad, velocidad, etc.).
- Se debe identificar a las funciones críticas. Debe utilizarse heramientas CASE para construir el modelo lógico (dentro de lo posible).

Actividades de la Fase

Especificación de Requisitos de Software:

- Requisitos funcionales.
- Requisitos de rendimiento.
- Requisitos de interfaces.
- Requisitos operacionales.
- Requisitos de recursos.
- Requisitos de verificación.
- Requisitos de tests de aceptación.
- Requisitos de documentación.
- Requisitos de seguridad de la información.
- Requisitos de transportabilidad.
- Requisitos de calidad.
- Requisitos de confiabilidad.
- Requisitos de mantención.
- Requisitos de seguridad de la operación.

Actividades de la Fase

Revisiones del SRD:

- Las salidas de esta fase deben ser revisadas formalmente durante la etapa de Revisión de Requisitos de Software (SR/R).
- Debiese ser una revisión técnica.
- Los participantes deben ser los usuarios, personal de operaciones, desarrolladores y administrador de proyectos.

Clasificación de Requisitos de Software

- ***Requisitos funcionales.***
  - Especifican qué debe hacer el software. Se derivan del modelo lógico.
- ***Requisitos de rendimiento.***
  - Establecen valores numéricos para variables medibles.
  - Pueden ser incluídos en la especificación cuantitativa de cada función, o especificados en forma independiente. Especificac. cualitativas no son aceptables.
  - Los atributos de rendimiento deben ser presentados como rangos de valores: peor caso aceptable; valor nominal a ser usado en la planificación; mejor caso, para indicar potencial de crecimiento.

Clasificación de Requisitos de Software

- ***Requisitos de interfaces.***
  - Especifican hardware, software o elementos de bases de datos con los que el sistema o sus componentes interactúan o se comunican.
  - Deben ser clasificados en requisitos de software, hardware y comunicaciones.
  - Puede utilizarse diagramas de bloques para ilustrar parte de este tipo de requisitos.
- ***Requisitos operacionales.***
  - Especifican la forma en que correrá el sistema y como se comunicará con los operadores humanos. Incluyen todas las interfaces de usuario, interacción humano-computador, y requisitos logísticos y organizacionales. Por ejemplo, se usará teclado/mouse.

Clasificación de Requisitos de Software

- ***Requisitos de recursos.***
  - Especifican los límites superiores de los recursos físicos tales como capacidad de procesamiento, memoria principal, espacio en disco, etc.
- ***Requisitos de verificación.***
  - Especifican las restricciones de cómo el software se verificará.
  - Pueden incluir requisitos de simulación, emulación, tests vivos con entradas simuladas, tests vivos con entradas reales, e interfaz con ambientes de testeo.
- ***Requisitos de tests de aceptación.***
  - Especifica las restricciones de cómo se validará el software.
  - Impone restricciones al SVVP (Software Validation and Verification Plan), si éste se construye (no es el caso de nuestro proyecto).

Clasificación de Requisitos de Software

- ***Requisitos de documentación.***
  - Especifica documentación adicional a la requerida en el estándar.
- ***Requisitos de seguridad de la información.***
  - Requisitos para asegurar el sistema contra amenazas a la confidencialidad, integridad y disponibilidad.
- ***Requisitos de transportabilidad.***
  - Especifican la facilidad para utilizar el sistema en otros computadores y sistemas operativos.
- ***Requisitos de calidad.***
  - Especifican atributos del software para asegurar que estará saludable para sus propósitos.Principalmente, las “ilidades del software”.
  - Definir atributos de calidad que sean claramente medibles.

Clasificación de Requisitos de Software

- ***Requisitos de confiabilidad.***
  - Especifican los tiempos medios entre falla aceptables.
- ***Requisitos de mantenibilidad.***
  - Especifica cuan fácil es reparar fallas y adaptar el software a nuevos requisitos.
  - Debiera especificarse en forma cuantitativa, tal como el tiempo medio para reparar una falla.
- ***Requisitos de seguridad de la operación.***
  - Especifican cualquier requisito para reducir la posibilidad de daño que puede producirse de una falla del software.

- La especificación de *atributos de requisitos* es igual a la del URD.

Completitud de Requisitos de Software

- Principalmente debe considerar dos aspectos:
  - ¿Se han considerado todos los requisitos de usuarios?
  - ¿Se ha especificado una actividad para validar cada conjunto posible de entradas?
- Se debe incluir una matríz de trazabilidad en el SRD para probar completitud.

Consistencia de Requisitos de Software

- Un conjunto de requisitos es consistente si y sólo si ningún subconjunto de requisitos entra en conflicto.
- Existe muchos tipos de inconsistencias:
  - términos diferentes para nombrar el mismo objeto.
  - El mismo término para nombrar diferentes objetos.
  - Actividades incompatibles pasando al mismo tiempo.
  - Actividades pasando en orden erróneo.

Duplicación de Requisitos de Software

- Debe ser evitado, aunque a veces es requerido para un mejor entendimiento del SRD.
- Cuando ocurran duplicaciones, debe hacerse referencias cruzadas para mejorar alteraciones.

Salidas de la Fase

- Documento de requisitos de software (SRD).
- Planes de testeo del sistema.
- El estándar establece otras salidas, pero nosotros no las consideraremos en nuestro proyecto.

Documento de Requisitos de Software

1. Introducción
   - Propósito
   - Alcance
   - Definiciones, acronismos y abreviaciones
   - Referencias
   - Resumen

… es igual a la introducción del URD. Sólo debe redefinirse o agregarse la información que se estime conveniente.

Documento de requisitos de software

2. Descripción General

- **Relación con proyectos actuales.** Establecer cuales son los proyectos, con los que está relacionado el software a construir. Establecer claramente la relación con cada uno de éstos (por ej. comparten los datos del “profesor”).
- **Relación con proyectos predecesores y sucesores.** Idem al anterior, pero para proyectos predecesores, y sucesores.
- **Función y propósito.** Narrativa que establece cuál es la función principal, y el propósito del sistema.
- **Consideraciones ambientales.** Narrativa que especifica aquellas consideraciones que deben ser tenidas en cuenta para el normal funcionamiento del software a desarrollar (por ej. “se debe realizar un backup diario”, o “la temp. no debe superar los 35ºC”).

Documento de requisitos de software

2. Descripción General

- **Relación con otros sistemas.** Establecer cuales son los sistemas existentes, con los que está relacionado el software a construir. Establecer claramente la relación con cada uno de éstos (por ej. comparten la tabla “persona”, de la B.D. “personal”).
- **Restricciones generales.** Básicamente son las mismas del URD. Pueden incorporar las actualizaciones que hayan surgido al analizar los requisitos de software.
- **Descripción del modelo.** Narrativa que establece, a grandes rasgos, cómo va a funcionar el sistema. Representa el modelo lógico del sistema. Pueden utilizarse diagramas de bloques, u otro tipo de diagramas genéricos, si lo desea.

Documento de requisitos de software

3. Requisitos específicos
   - Requisitos funcionales
   - Requisitos de rendimiento
   - Requisitos de interfaces
   - Requisitos operacionales
   - Requisitos de recursos
   - Requisitos de verificación
   - Requisitos de aceptación de tests
   - Requisitos de documentación
   - Requisitos de seguridad de la información
   - Requisitos de transportabilidad
   - Requisitos de calidad
   - Requisitos de confiabilidad
   - Requisitos de mantensión
   - Requisitos de seguridad de la operación
4. Matriz de trazabilidad de requisitos de usuarios vs requisitos de software

Administración de Requisitos durante un Proyecto

- Los documentos URD y SRD deben ser “documentos completos”. Esto significa que todos los requisitos conocidos deben ser incluídos cuando son producidos.
- Sin embargo, es altamente probable que aparezcan nuevos requisitos después de las aprobaciones del URD y SRD.
- Por ello, es necesario establecer al inicio del proyecto, procedimientos para manejar nuevos requisitos.
- Es necesario no comprometer la integridad del diseño cuando se incorporan nuevos requisitos. Dichos requisitos, si es que son aceptados tanto por el usuario como por el diseñador, debiesen ser manejados de la misma forma que los requisitos originales.

Administración de Requisitos durante un Proyecto

- El procedimiento para incorporar un nuevo requisito de usuario es el siguiente:
  - Generar un nuevo manuscrito del URD.
  - Convenir una revisión de los UR, y si el cambio es aceptado,
  - Repetir las fases SR, AD, y DD para incorporar el nuevo requisito y sus consecuencias.
- El procedimiento para incorporar un nuevo requisito de software es similar.

Administración de Requisitos durante un Proyecto

- Otra forma de manejar nuevos requisitos es el de instituir un Comité de Revisión de Software después de UR/R en vez de DD/R.
- Otra forma es el uso de un enfoque de ciclo de vida de desarrollo evolutivo. Sin embargo, esto sólo alarga la administración de los nuevos requisitos para el release posterior al que está en preparación, lo que puede no ser suficiente.

Administración de Requisitos durante un Proyecto

- La calidad del trabajo hecho para las fases UR y SR puede ser medido por el número de requisitos que aparecen en fases posteriores.
- Especial énfasis debe dársele a la aparición de nuevos requisitos. Un número de apariciones importantes es un signo claro que el software muy probablemente no será un éxito.

Administración de Requisitos durante un Proyecto

- La disponibilidad de herramientas puede ser crítica para un proyecto con cambios de requisitos frecuentes.
  - En proyectos donde se conviene en congelar los requisitos en la fase SR, el uso de métodos basados en papel para realizar análisis de requisitos y especificación de diseño pueden ser suficientes.
  - En proyectos donde no es posible congelar los requisitos, puede ser escencial el uso de herramientas de ingeniería de software que permitan que nuevos requisitos y cambios al diseño sean asimilados rapidamente para evitar atrasos serios.

Bibliografía y Casos

- I. Sommerville, P. Sawyer. Requirements Engineering, A Good Practice Guide.Wiley, 1999.
- S. Robertson, J. Robertson. Mastering the Requirements Process. Addison-Wesley, 1999.
- A. Davis. Software Requirements, Revision. Prentice Hall, 1993.
- ESA, SOFTWARE ENGINEERING STANDARDS, ISSUE 2. Preparado por: ESA Board for Software Standardisation and Control (BSSC). URL: http://www.estec.esa.nl/wmwww/WME/index.html
- R. L. Kliem, Collecting Project Information.
- R. H. Deane, Linking Project Outcomes to Customer Needs.