



CENTRO DE ESTUDIOS FINANCIEROS

VIRIATO, 52	28010 MADRID	914 44 49 20
PONZANO, 15	28010 MADRID	914 44 49 20
G. DE GRÀCIA, 171	08012 BARCELONA	934 15 09 88
ALBORAYA, 23	46010 VALENCIA	963 61 41 99

www.cef.es

info@cef.es

Índice Tema 4

1. El análisis del sistema. Ideas generales.
 - 1.1. Problemas y principios fundamentales del análisis.
 - 1.1.1. El dominio de la información.
 - 1.1.2. El principio de la partición.
 - 1.1.3. Las visiones lógica y física del sistema.
 - 1.2. Alcance del análisis.
2. La fase de requisitos del software. Conceptos generales.
3. La identificación o educación de los requerimientos del software.
 - 3.1. Principales estrategias para la obtención de requisitos.
 - 3.1.1. La Técnica JAD.
 - 3.1.2. El prototipado.
 - 3.1.3. El Análisis de los Factores Críticos de Éxito.
 - 3.1.4. Las Entrevistas.
4. Análisis y negociación de los requisitos identificados.
 - 4.1. Técnicas de análisis.
5. La documentación de los requisitos.
 - 5.1. Especificación de los requisitos funcionales.
 - 5.2. Especificación de los requisitos no funcionales.
6. La visión de los requisitos según la metodología métrica v.3.
 - 6.1. Obtención de requisitos.
 - 6.2. Especificación de Casos de Uso.
 - 6.3. Análisis de Requisitos.
 - 6.4. Validación de Requisitos.



CENTRO DE ESTUDIOS FINANCIEROS

VIRIATO, 52	28010 MADRID	914 44 49 20
PONZANO, 15	28010 MADRID	914 44 49 20
G. DE GRÀCIA, 171	08012 BARCELONA	934 15 09 88
ALBORAYA, 23	46010 VALENCIA	963 61 41 99

www.cef.es

info@cef.es

TEMA 4

Estrategias de determinación de requerimientos: entrevistas, derivación de sistemas existentes, análisis y prototipos.

INTRODUCCIÓN.

El proceso de desarrollo de cualquier sistema de información consta de tres fases genéricas: definición, desarrollo y mantenimiento, con independencia de cuál sea el área de aplicación, el tamaño del proyecto, su complejidad y el modelo de ciclo de vida elegido.

El objeto de la fase de definición del sistema es determinar con claridad y precisión qué es lo que hay que hacer y para ello será necesario identificar los requerimientos clave del sistema en su totalidad (Análisis global del Sistema) y del software en particular (Análisis de Requerimientos del Software).

En palabras de Brooks: «Lo más difícil en la construcción de un sistema software es decidir qué construir. No existe ninguna tarea con mayor capacidad de lesionar al sistema cuando se hace mal, y ninguna tarea es tan difícil de rectificar *a posteriori*».

De la importancia de los requisitos en la Ingeniería del Software basten algunos datos como muestra. Algunos estudios reflejan que el 45 por 100 de los errores tienen su origen en los requisitos y en el diseño preliminar, otros señalan que el 56 por 100 de los errores que tienen lugar en un proyecto software se deben a una mala especificación de requisitos, etc.

Los factores principales que conducen al fracaso en los proyectos software son: la falta de comunicación con los usuarios, los requisitos incompletos y los cambios en los requisitos; y la evidencia demuestra que los requisitos contienen demasiados errores, que muchos de estos errores no se detectan al principio, pero podrían ser detectados, y que no detectar estos errores incrementa grandemente los costes del proyecto y su duración. La consecuencia es que el sistema no satisfará a los usuarios, se producirán desacuerdos entre usuarios y desarrolladores, y se gastará tiempo y dinero en construir un sistema equivocado.

Quizá no sea posible elaborar los requisitos con absoluta perfección, pero se debe intentar minimizar el impacto de los errores en los requisitos y organizar mejor las tareas relacionadas con ellos. Para remediar esta situación surge la Ingeniería de Requisitos, que trata de los principios, métodos, técnicas y herramientas que permiten descubrir, documentar y mantener los requisitos para sistemas basados en computadora, de forma sistemática y repetible.

El objeto de este tema es, precisamente, el estudio de la Ingeniería o Fase de Requisitos.

Dado que esta fase se encuadra dentro de lo que se viene conociendo por análisis, primeramente haremos referencia a esta actividad, a fin de dejar claro en qué consiste el análisis, qué problemas plantea, en qué principios generales se basa y cuál es su alcance.

Seguidamente nos centraremos en las actividades propias de la Ingeniería de Requisitos, presentando los conceptos generales a tener en cuenta y entrando a continuación en el estudio detallado de cada actividad de la fase de Requisitos, esto es:

- Identificación de los requisitos, explicando algunas técnicas para la obtención de los mismos, especialmente la técnica de las entrevistas.
- Análisis y negociación de los requisitos identificados, mencionando las principales técnicas de análisis, que no se abordarán en este tema por ser objeto de otros posteriores.
- Documentación de los requisitos, tanto funcionales como no funcionales, explicando el contenido del documento donde se han de plasmar éstos, el documento de Especificación de Requisitos Software (ERS).

Finalmente se hará una referencia a la consideración de los requisitos en la metodología Métrica v.3.

1. EL ANÁLISIS DEL SISTEMA. IDEAS GENERALES.

En Ingeniería del Software, el término «análisis» posee dos acepciones: por una parte, se refiere a una determinada fase del proceso de desarrollo de un sistema de información, y por otra, al conjunto de productos que se originan como resultado de esa fase. El acuerdo entre los distintos autores termina en este punto. A partir de aquí, cada uno establece objetivos distintos para la fase de análisis y, por lo tanto, un distinto conjunto de tareas a realizar y un distinto conjunto de productos de salida obtener.

Como ejemplo de la confusión existente, la siguiente tabla muestra el uso de una terminología diferente durante las primeras fases del desarrollo.

ACTIVIDAD	AUTORES					
	BÖEHM 1976 IEEE 1984	ROSS 1977	FREEMAN 1983	BERZINS 1985	YOURDON 1989	DAVIDS 1993
Análisis del problema	Requisitos	Análisis del contexto	Análisis	Definición de requisitos	Modelo esencial	Análisis del problema
Definición del comportamiento externo		Especific. funcional	Especific. funcional	Especific. funcional	Modelado de la implementación de usuario	Especificación
Definición de los componentes del producto	Diseño	Diseño		Diseño	Diseño	Diseño preliminar

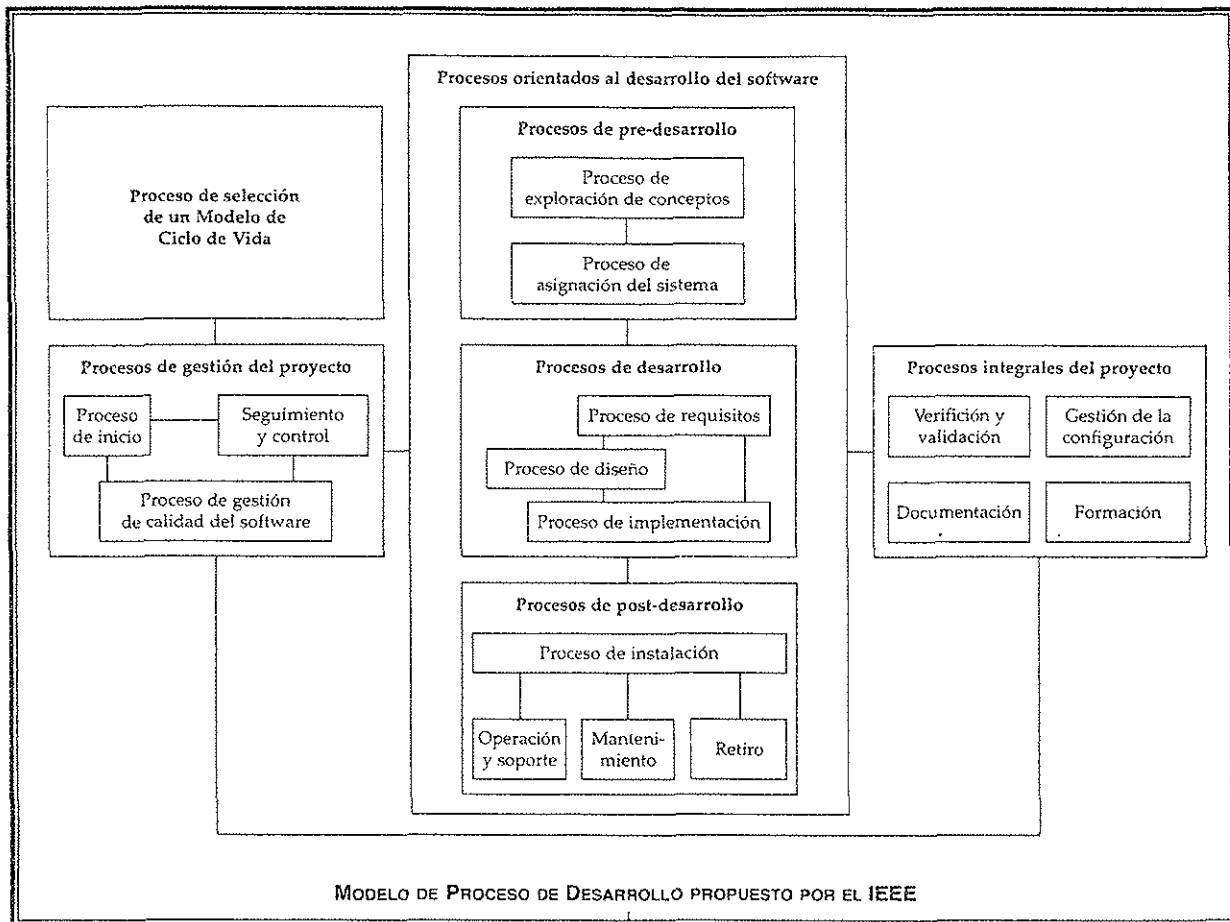
La situación actualmente, lejos de mejorar, se ha complicado. El término «análisis» prácticamente ha desaparecido de la terminología informática de uso corriente y ha sido sustituido por el término «Ingeniería de Requisitos», posiblemente más nebuloso.

En cualquier caso, con independencia de la nomenclatura que se quiera emplear, el análisis es la actividad que incluye aprender acerca del problema a resolver, entender las necesidades de los potenciales usuarios, tratar de identificar quién es realmente el usuario y entender todas las restricciones a la solución.

Dos palabras son claves en la definición anterior: aprender y entender. El análisis pretende proporcionar el conocimiento necesario para solucionar los problemas que surgen en un dominio determinado, y ese conocimiento a adquirir sólo puede obtenerse mediante una inmersión en el dominio del problema, el cual pertenece a los clientes o usuarios.

La actividad de análisis tiene por objetivo proporcionar los conceptos y restricciones relevantes para el desarrollo del futuro sistema software, y es una parte integrante del proceso de desarrollo. Según el Modelo de Proceso de Desarrollo propuesto por el IEEE, la actividad de análisis está oculta y parcialmente imbricada en los siguientes tres procesos (véase la siguiente figura) aunque no coincide exactamente con ninguno:

1. Exploración de conceptos. Se realiza una exploración inicial de los conceptos relevantes del problema y de las restricciones de la solución, con el objetivo de identificar una configuración (hardware, software, etc.) que resuelva el problema.
2. Asignación del sistema. Se identifican los componentes de la solución, es decir, del sistema, y se asigna al software una parte del problema a fin de poder iniciar el posterior proyecto de desarrollo. Este proceso tiene fuertes características de evaluación de alternativas y toma de decisión.
3. Proceso de requisitos. Centrándose ya exclusivamente en la parte software del sistema, aquí se identifican los requisitos que éste debe cumplir. Algunos requisitos vendrán dados por la asignación realizada en el proceso anterior, pero muchos otros quedarán todavía por descubrir y definir.



En definitiva, el término «Análisis del Sistema» puede inducir a confusión si sólo se usa en el contexto que alude a las actividades del análisis de requerimientos del software. En general, puesto que un sistema comprende algo más que el elemento software, el Análisis del Sistema abarca todos sus componentes.

Una parte del análisis (la exploración de conceptos y la asignación del sistema), que para nuestro entendimiento podríamos denominar «Análisis Global», se llevará a cabo previamente al desarrollo propiamente dicho del software, por lo que la podemos englobar genéricamente dentro de la categoría de procesos de pre-desarrollo, y cuyo objeto es obtener una especificación general que sirva de punto de partida a todo el desarrollo en sí del sistema.

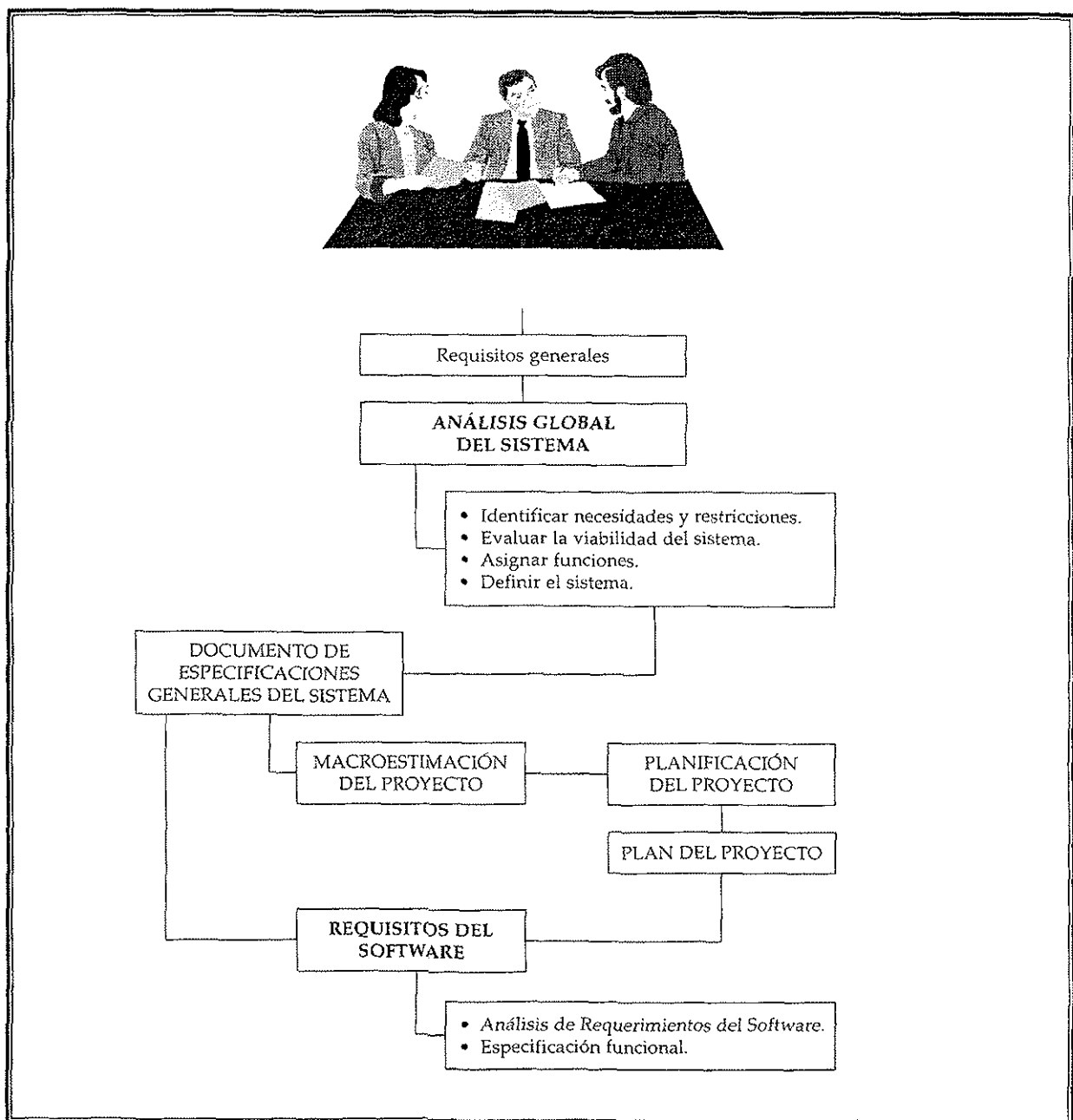
Concretando más, los objetivos que persigue este Análisis Global son:

- Identificar las necesidades del cliente y establecer las restricciones de coste y tiempo que incidirán en el futuro desarrollo.
- Evaluar la viabilidad del sistema, tanto desde el punto de vista técnico, como desde el punto de vista económico y legal.
- Asignar las funciones que dentro del sistema ha de cumplir el elemento hardware, el software y el elemento humano.
- Crear una definición clara y precisa del sistema, que sea la base para todo lo que sigue después.

Asociados a tales objetivos, las tareas que comprende el Análisis del Sistema son:

1. La identificación de necesidades y restricciones.
2. El estudio y evaluación de la viabilidad del sistema.
3. La asignación de funciones y compromisos.
4. La especificación del sistema.

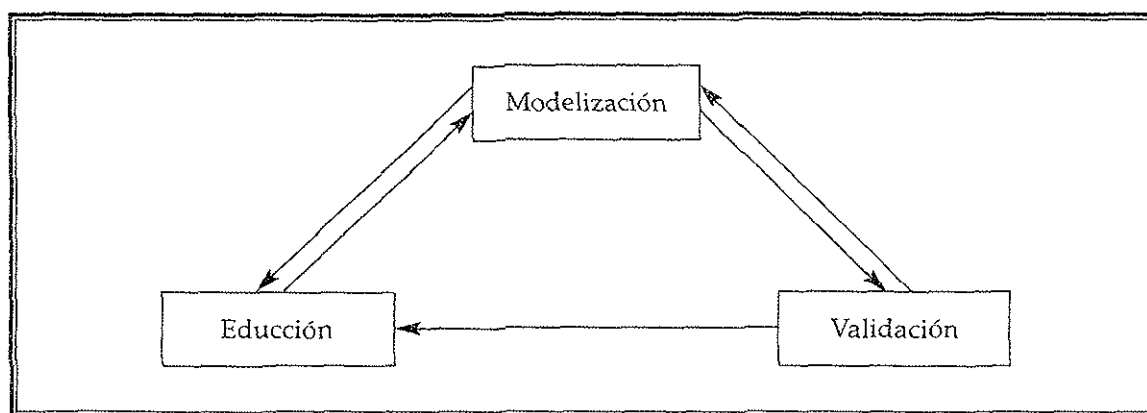
La otra parte del análisis (proceso de requisitos) se centra ya en el componente software del sistema y es, por así decirlo, el primer proceso de desarrollo propiamente dicho.



Por lo que atañe a los requisitos del sistema software, en los últimos años se ha desarrollado una gran actividad investigadora en el campo de la «Ingeniería de Requisitos», el cual incluye de forma natural al análisis, dando como resultado la identificación de dos aproximaciones distintas para la realización de la actividad de análisis: la aproximación clásica y la aproximación propuesta por el SWEBOK.

Según la aproximación clásica, el análisis consta de tres tareas:

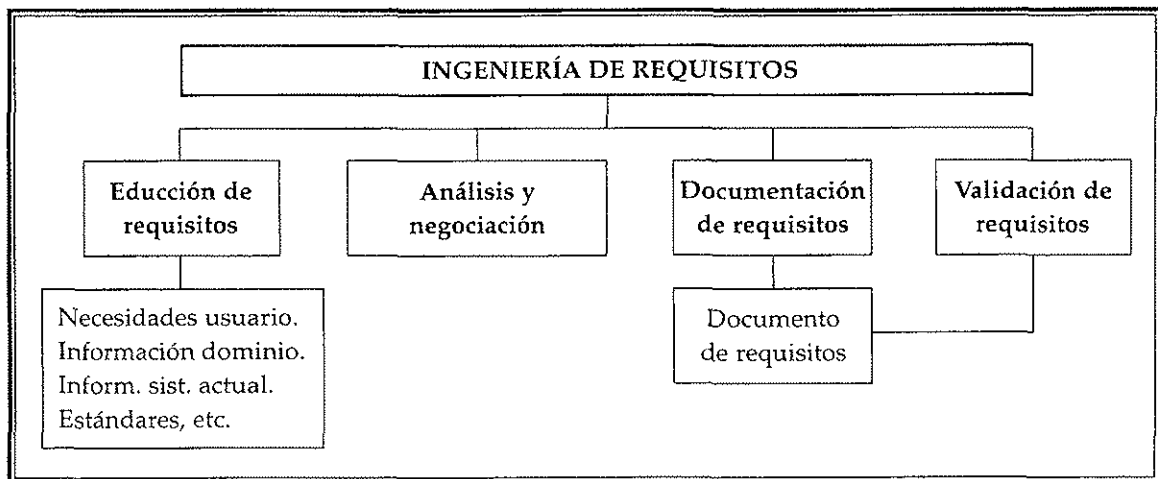
- Educación, cuyo objetivo es adquirir el conocimiento del dominio de los clientes y usuarios, de forma que sea posible identificar los conceptos, relaciones y funciones más relevantes. Esta tarea tiene características de investigación y es difícilmente formalizable. En Ingeniería del Software se utilizan una serie de técnicas para realizar esta tarea como, por ejemplo, las entrevistas, el análisis de documentos, el prototipado y el JAD (Joint Application Development).
- Modelización, cuyo objetivo es representar, mediante la utilización de Modelos Conceptuales, los conocimientos adquiridos en la tarea anterior. Los modelos conceptuales son mecanismos de representación que permiten registrar los conocimientos adquiridos durante la educación con el fin de facilitar su comprensión y permitir su comunicación entre todos los participantes en la actividad de análisis (clientes, usuarios y analistas).
- Validación, cuyo objetivo es verificar la exactitud de los conocimientos adquiridos.



La aproximación más moderna, propuesta por el SWEBOK (Software Engineering Body of Knowledge) propone un conjunto de cuatro tareas para la fase completa de Ingeniería de Requisitos. Éstas son:

- Educación de requisitos, que tiene el mismo alcance que en la aproximación clásica.
- Análisis y negociación, que engloba las siguientes subtareas:
 - Clasificación de los requisitos identificados, a fin de facilitar su comprensión y su utilización en posteriores actividades del desarrollo.
 - Modelado conceptual, con los mismos objetivos que en la aproximación clásica, salvo que en vez de trabajar con conocimiento del dominio, está trabajando con requisitos del sistema.

- Negociación, cuyo objetivo es solucionar los problemas que puedan surgir debido a conflictos entre requisitos.
- Documentación de requisitos, cuyo objetivo es estabilizar los requisitos identificados y sin conflictos en un documento que sirva de referencia para todo el proceso de desarrollo.
- Validación, con el mismo alcance que en la aproximación clásica.



1.1. PROBLEMAS Y PRINCIPIOS FUNDAMENTALES DEL ANÁLISIS.

El Análisis supone una actividad intensiva de comunicación entre el analista y el cliente o usuario. Los problemas pueden surgir cuando esta comunicación se interrumpe, carece de fluidez o tiene disfunciones por cualquier otra causa; por tanto, es importante que el analista exhiba determinados rasgos tales como:

- Capacidad de comunicación.
- Capacidad de síntesis de los problemas.
- Capacidad de comprensión de conceptos abstractos.
- Capacidad para entresacar lo importante de fuentes confusas.
- Capacidad de captación de los problemas del entorno de usuario.
- Habilidad para evitar que «los árboles no dejen ver el bosque». Es decir, no perder de vista el objeto global de los programas.

Entre los problemas que pueden encontrarse durante el Análisis están las dificultades asociadas a la adquisición de la información pertinente, el manejo de la complejidad del problema y el acomodar los cambios que puedan ocurrir durante y después del análisis. Conforme crece el tamaño del problema, crece la complejidad del Análisis ya que cada nuevo elemento, función o ligadura puede tener efecto sobre los otros elementos del sistema, por esta razón, el trabajo de análisis crece geométricamente con la complejidad del problema.

Los problemas subyacentes al Análisis son atribuibles a muchas causas, entre otras:

- Pobre comunicación, lo que hace difícil la adquisición de información.
- Uso de técnicas y herramientas inadecuadas, lo que produce especificaciones malas o imprecisas.
- Tendencia a acortar la duración del análisis, lo que conduce a un análisis inestable.

A efectos de reducir los problemas y la complejidad que en sí conlleva la actividad de análisis, se pueden emplear tres técnicas o principios, que son en esencia herramientas para el manejo y reducción de la complejidad. Éstas son la abstracción, la partición y la proyección.

- a) La técnica de abstracción consiste en sustituir una cosa, o un conjunto de cosas, por otra que resume o resume a las cosas sustituidas. El objetivo de la abstracción es controlar la complejidad al reducir el detalle estructural o funcional de las cosas en sí, sustituyéndolas por otras más sencillas de manejar intelectualmente. La abstracción define la relación «general/específico» o «ejemplo de», entre objetos, funciones o estados en el dominio del problema.
- b) La partición es una técnica de descomposición cuyo objetivo es dividir un problema complejo, ya sea estructuralmente o funcionalmente, en varios subproblemas y, de esta forma, repartir la complejidad inherente al problema inicial entre los distintos subproblemas. La partición define la relación «agregación/parte de», entre objetos, funciones o estados en el dominio del problema.
- c) La técnica de proyección tiene por objeto «ver» un problema desde distintas perspectivas o puntos de vista. La partición define la relación «visión de», entre objetos, funciones o estados en el dominio del problema.

Además de las técnicas anteriores, y por lo que se refiere ya a los requisitos del sistema software, aunque no sea una panacea, la aplicación de unos principios fundamentales y de unos métodos sistemáticos de análisis reducirá grandemente el impacto de los problemas asociados al mismo. La multiplicidad y disparidad de métodos de análisis (todas las metodologías con sus diversas herramientas) obedecen a tres principios fundamentales en los cuales se fundamenta todo el proceso que se lleva a cabo en la fase de Requisitos del Software. Estos principios son:

1. El dominio de la información y el dominio funcional de un problema deben ser representados y comprendidos.
2. Principio de la partición: el problema debe subdividirse, de forma que los detalles se vayan descubriendo de una manera progresiva o jerárquica.
3. Deben desarrollarse las representaciones lógicas y físicas del sistema.

1.1.1. El dominio de la información.

El dominio de la información hace referencia a los datos que han de ser procesados por los programas que constituyen el software. Las visiones bajo las que se ha de representar el dominio de la información son las tres siguientes:

- a) Flujo de la Información: es la visión que se representa la manera en que cambian los datos conforme pasan a través del sistema a lo largo de un camino de transformación. Las transformaciones que se aplican a los datos son las funciones o subfunciones que un programa debe ejecutar.
- b) Contenido de la Información: es la visión que se representa los elementos de datos individuales que componen otros elementos mayores de información. Por ejemplo, los campos o atributos que integran los registros.
- c) Estructura de la Información: es la visión que representa la organización lógica de los distintos elementos de datos. Por ejemplo, la organización de los ficheros o entidades de datos.

1.1.2. El principio de la partición.

Normalmente los problemas que se pretenden abordar mediante la realización de un sistema software son demasiado grandes y complejos para ser comprendidos como un todo. Por esta razón conviene particionar o subdividir tales problemas en partes que puedan ser fácilmente comprendidas y establecer interfaces entre las partes, de forma que se realice la función global que da solución al problema.

La partición del problema, en esencia, descompone éste en sus partes constituyentes. Conceptualmente, se establece una representación jerárquica de la función o de la información y luego se subdivide el elemento superior primero a nivel horizontal, descomponiendo funcionalmente el problema, y después a nivel vertical, incrementando los detalles para cada función.

1.1.3. Las visiones lógica y física del sistema.

La visión lógica del sistema presenta las funciones que ha de realizar éste y la información que ha de procesar, con independencia de los detalles de implementación. Una representación de los requerimientos del software es imprescindible para un correcto diseño.

La Fase de Requerimientos del software debe enfocarse sobre lo que el software debe realizar y no sobre cómo se efectuará la implementación. En esta línea, Mc Menamin y Palmer han propuesto un método de moderación para separar la esencia del sistema de los detalles de su implementación:

- El modelo esencial o modelo lógico, el cual describe lo que un sistema debe realizar e incluye descripciones de las funciones y del contenido y estructura global de los datos.
- El modelo de materializaciones o modelo físico, que suministra información sobre las conexiones de hardware, el acoplamiento del sistema operativo específico y demás detalles relativos a la implementación.

1.2. ALCANCE DEL ANÁLISIS.

Uno de los problemas de la actividad de análisis es que se sabe cuándo empieza, pero es muy difícil saber cuándo acaba. El problema de cuándo termina la actividad de análisis está relacionado con la determinación de qué es análisis y qué es diseño.

La actividad de análisis es plenamente conceptual y su propósito fundamental es conseguir el entendimiento del dominio de los clientes o usuarios. Desgraciadamente, el diseño tiene casi los mismos objetivos, si bien referidos a un sistema basado en computadora y no al dominio de los clientes o

usuarios. Aún más, las herramientas utilizadas en el diseño son, en gran medida, similares a las utilizadas durante el análisis. Por esta razón análisis y diseño se confunden a veces y sus objetivos se mezclan. Una regla que puede tenerse en cuenta a fin de evitar este problema es la siguiente:

«El análisis debe terminar cuando los conceptos del dominio estén lo suficientemente claros como para que el analista sea capaz de idear una solución software para el problema bajo estudio».

2. LA FASE DE REQUISITOS DEL SOFTWARE. CONCEPTOS GENERALES.

Una vez aprobadas por el cliente o usuario las «Especificaciones Generales del Sistema» como punto final del análisis global del mismo, y obtenida y aprobada, en base a las especificaciones generales, una primera planificación del proceso de desarrollo del sistema, esto es el «Plan del Proyecto», comienza, verdaderamente, el proceso de la Ingeniería del Software.

Antes de seguir adelante conviene matizar, primeramente, el alcance del término «requisitos», pues no tiene el mismo significado para el cliente o usuario que para el desarrollador del software. Mientras que para el usuario, los requisitos son las condiciones o las capacidades necesarias que debe reunir un sistema en general y el software en particular, para que dicho usuario pueda resolver un problema o alcanzar un objetivo, para el equipo de desarrollo del software, los requisitos son las capacidades o condiciones que debe reunir un sistema para satisfacer un contrato estándar o cualquier otro documento impuesto formalmente.

En cualquier caso, si consideramos que todo problema software consiste en configurar una máquina M , que tendrá un comportamiento externo M_{ex} y un comportamiento interno M_{in} , para que satisfaga unas necesidades, metas y objetivos (requisitos) R en un contexto o dominio D , por extensión, en Ingeniería de Requisitos, se denominan «requisitos» al conjunto $\{M_{\text{ex}}-R-D\}$, aunque tan sólo R sean propiamente requisitos. La razón de esta definición es que los «requisitos» están constituidos por todas aquellas informaciones necesarias para construir un sistema que satisfaga las metas perseguidas.

La fase de Requisitos del Software es el último eslabón de la fase de definición de todo proceso de desarrollo del software y, a la vez, es el primer paso técnico en el proceso de la Ingeniería del Software. Abarca el proceso de comprensión de los requerimientos de un sistema software y proporciona, de forma general, el ámbito del software y una especificación concreta del mismo, la cual se convierte en la base de la futura fase de desarrollo, propiamente dicha.

Concretando más, los objetivos que persigue la fase de Requisitos del Software son:

- Facilitar la especificación de la funcionalidad y el comportamiento del software.
- Indicar las interfaces del software con otros elementos del sistema.
- Establecer las ligaduras de diseño que deben cumplir los programas que conforman el software.

Siguiendo la aproximación propuesta por el SWEBOK, en la que se apoya la Ingeniería de Requisitos, la fase de Requisitos del Software comprende cuatro actividades específicas:

- a) La identificación o educación de requisitos, que es el proceso a través del cual, tanto el cliente o el usuario, como el ingeniero de software, a partir de las «Especificaciones Generales del Sistema», del «Plan del Proyecto» y de otras fuentes, descubren, revelan, articulan y comprenden

los requerimientos del software.

- b) El análisis y negociación de los requisitos, que es el proceso de evaluar y sintetizar los requisitos que han sido identificados, en orden a detectar problemas de inconsistencias entre los mismos, inconcreciones, etc. Durante esta etapa se evaluará el flujo y la estructura de la información, refinando en detalle todas las funciones que ha de cumplir el software y estableciendo las características de la interface del sistema y las ligaduras de diseño.
- c) La documentación o representación de los requisitos, que es el proceso de registrar los mismos en un documento, a través de una o varias formas como: el lenguaje natural, el simbólico, representaciones gráficas, etc. Es decir, es la especificación formal de los requisitos.
- d) La validación de los requisitos, que es el proceso de establecer los criterios y técnicas que aseguran que el software, cuando se produzca, cumple con los requisitos. Asimismo, como punto final de esta etapa y de la fase de Requisitos del Software, el documento que genéricamente podríamos denominar como «Especificación de Requisitos del Software» será revisado y aprobado por el cliente o usuario.

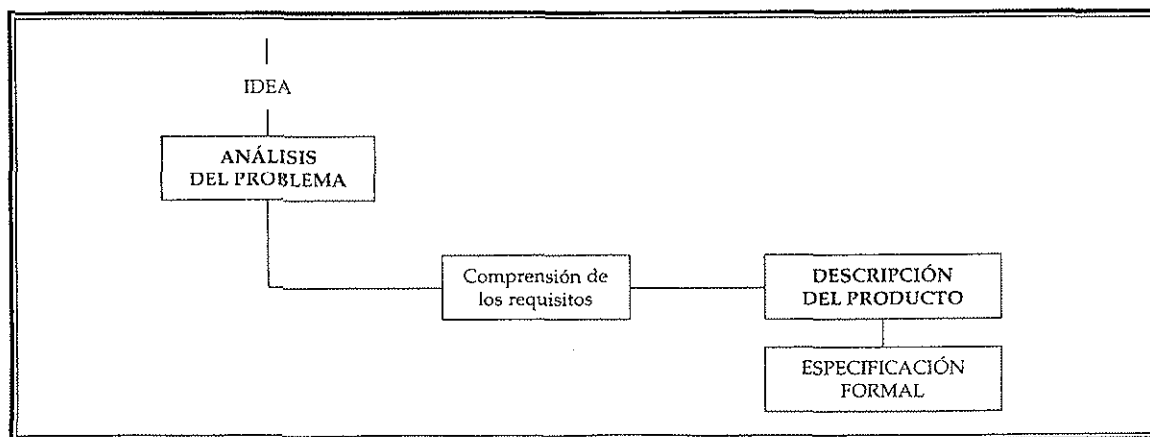
Es importante destacar que estas cuatro actividades no pueden separarse estrictamente y realizarse secuencialmente, sino que están entrelazadas y se llevan a cabo repetidamente.

La mayoría de los autores y textos de Ingeniería del Software no utilizan, generalmente, esta terminología completa que acabamos de exponer, sino que la simplifican. Así, consideran que la fase de Requisitos del Software se puede descomponer en dos actividades o subprocesos:

1. Análisis de Requisitos o de Requerimientos del Software, también llamada, según algunos textos, Análisis del Problema, Definición o Especificación de Requisitos o Modelización Esencial.
2. Especificación Funcional, también llamada: Descripción del Producto, Definición del Comportamiento Externo o Escritura de la Especificación de Requisitos.

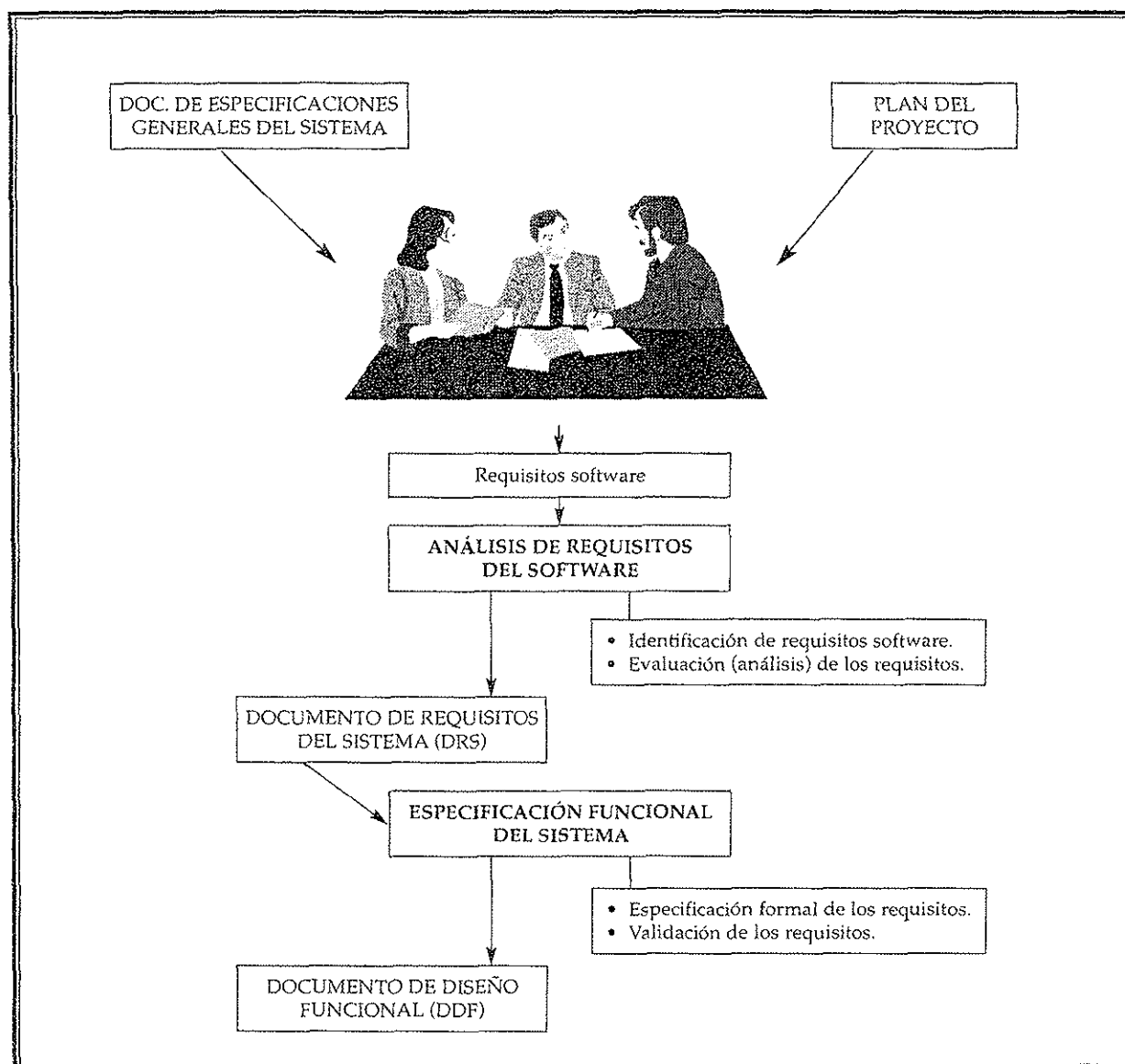
Durante el Análisis de Requisitos del Software, los analistas adquieren todo el conocimiento posible del problema, identificando las posibles restricciones a la solución del mismo. Es decir, llegan a una comprensión total de los requisitos que ha de cumplir el software.

Durante la Especificación Funcional, los analistas describen, especifican y representan en un documento el comportamiento esperado del producto para resolver el problema.



Comparando esta terminología clásica con la primera que hemos utilizado, parece claro que el Análisis del Problema o Análisis de Requerimientos del Software engloba las actividades de «identificación de requisitos» y «análisis y negociación», mientras que la Especificación Funcional comprende la «representación de los requisitos» y la «validación» de los mismos.

La siguiente figura representa esquemáticamente la fase de Requisitos del Software, indicando los documentos a obtener.



El Documento de Requisitos del Sistema (DRS) mostrará de forma ordenada el catálogo de requisitos que ha de cumplir el software del sistema, con sus prioridades. Este documento es la consecuencia de la identificación de los requisitos del software y de su evaluación, y es el punto de partida, una vez aprobado por el cliente o usuario, para la especificación formal del sistema.

El Documento de Diseño Funcional (DDF) es en el que verdaderamente se escribe la especificación formal del sistema, ya sea a través de representaciones gráficas, lenguaje natural, simbólico, u otros procedimientos. Su importancia es fundamental, ya que es, prácticamente, el último documento que será revisado y aprobado por el cliente o usuario y, además, es el punto de partida para la siguiente fase de desarrollo del software: la fase de diseño.

Generalmente, el Documento de Requisitos del Sistema (DRS) se suele integrar en el Documento de Diseño Funcional (DDF) y constituir así, ambos, un único documento que podríamos denominar «Documento de Especificación de Requisitos del Software».

3. LA IDENTIFICACIÓN O EDUCIÓN DE LOS REQUERIMIENTOS DEL SOFTWARE.

Al igual que en el Análisis global del Sistema la primera actividad consistía en identificar los requisitos del mismo, en la fase de Requisitos del Software, la primera actividad también es la educación o identificación de los requerimientos del software.

El término «educir» significa sacar u obtener algo de diversas fuentes. Por tanto, la educación, identificación o determinación de requisitos es el paso durante el cual los requisitos del software son obtenidos de fuentes tales como: la gente implicada (usuarios, clientes, expertos en la materia, etc.), las necesidades que ha de satisfacer el sistema, el entorno físico que rodea al sistema, el entorno organizacional, etc.

Si la fuente de la que se van a obtener los requisitos del software es la gente implicada en el sistema, el procedimiento general a seguir para la identificación de dichos requisitos consta de los siguientes pasos:

1. Identificar las fuentes relevantes, esto es, qué usuarios pueden proporcionar información acerca de los requisitos.
2. Preguntar las cuestiones apropiadas para obtener y comprender sus necesidades.
3. Analizar la información reunida buscando implicaciones, inconsistencias o aspectos no resueltos.
4. Confirmar con los usuarios la comprensión de los requisitos.
5. Sintetizar las especificaciones adecuadas para los requisitos.

Los problemas de la obtención de requisitos se pueden agrupar en tres categorías:

- a) Problemas de alcance, ya que los requisitos pueden implicar demasiada o muy poca información.

La solución a este problema es determinar claramente el contexto del sistema, es decir, los límites y objetivos del mismo, a fin de comprobar con ellos que los requisitos software obtenidos son correctos. Si no se contempla el contexto donde va a funcionar el sistema, se puede llegar a requisitos incompletos, no verificables, innecesarios y no utilizables.

- b) Problemas de comprensión, como consecuencia de una pobre comunicación entre usuario y analista. En este caso los requisitos obtenidos son ambiguos, incompletos, inconsistentes e incorrectos, porque no responden a las verdaderas necesidades de los usuarios.

La solución es que exista una buena comunicación entre usuarios, desarrolladores y clientes, a fin de que los requisitos se puedan escribir de manera que permitan, tanto que el desarrollador pueda distinguir si dichos requisitos se pueden implementar, como que el personal de control pueda comprobar si la implementación cumple con los requisitos.

- c) Problemas de volatilidad, ya que los requisitos evolucionan con el tiempo. En efecto, a medida que avanza el desarrollo del sistema, las necesidades del usuario pueden madurar a causa del conocimiento adicional fruto del desarrollo, o de necesidades del entorno o de la organización no previstas.

La solución es incorporar estos cambios a los requisitos originales, pues si no, éstos serán incompletos e inconsistentes con la nueva situación.

En definitiva, debido a los problemas de alcance, comprensión y volatilidad de los requisitos, puede ocurrir que las necesidades de los usuarios no estén expresadas claramente en un principio y, por tanto, que el analista asuma algunos puntos incorrectos. La solución será considerar el proceso de educación o identificación de requisitos como un proceso iterativo y emplear una serie de técnicas de obtención, expresamente diseñadas para minimizar los problemas anteriores.

3.1. PRINCIPALES ESTRATEGIAS PARA LA OBTENCIÓN DE REQUISITOS.

Son ciertamente numerosas las estrategias y técnicas que se han desarrollado para la obtención o educación de los requisitos, con el objetivo claro de superar los problemas de alcance, comprensión y volatilidad de los requisitos que se acaban de describir.

Según el entorno que se pretenda abarcar, las técnicas de educación se clasifican en:

- Técnicas de alto nivel, las cuales abarcan amplios entornos de trabajo que permiten albergar procesos que educen requisitos. Algunas técnicas de alto nivel son:
 - La técnica JAD (Joint Application Design).
 - El prototipado.
 - El Análisis de los Factores Críticos de Éxito.
- Técnicas de bajo nivel, las cuales proporcionan tácticas específicas que permiten obtener detalles de una parte determinada del sistema o de un usuario particular. Algunas técnicas de bajo nivel son:
 - Las Entrevistas.
 - El Brainstorming, que es una técnica de grupo sencilla para la generación de ideas. En un ambiente distendido y libre de críticas, la gente sugiere ideas.

Basándose en la actividad del analista, las técnicas de educación u obtención de requisitos se pueden clasificar en las siguientes categorías:

- Preguntar.
- Observar e inferir.
- Discutir y formular.
- Negociar con respecto a un conjunto estándar.
- Estudiar e identificar problemas.
- Descubrir a través de procesos evolutivos.
- Postular.

Evidentemente, todas las técnicas, ya sean de alto nivel o detalladas, participan de varias actividades del analista. Así, por ejemplo, el prototipado exige preguntar, discutir y formular, estudiar e identificar problemas y descubrir a través de procesos evolutivos.

3.1.1. La Técnica JAD.

Las sesiones JAD (Joint Application Design) tienen como objetivo reducir el tiempo de desarrollo de un sistema manteniendo la calidad del mismo. Para ello se involucra a los usuarios a lo largo de todo el desarrollo del sistema, es decir, desde la identificación de la necesidad, la propuesta de alternativas de solución y sobre todo en la especificación de los requisitos que debe cubrir el sistema y en la validación de prototipos. En definitiva, JAD es una técnica para promover la cooperación, comprensión y equipo de trabajo entre clientes, usuarios y desarrolladores, proporcionando un proceso que facilita la creación de una visión compartida de lo que el sistema debería ser.

Esta técnica implica a desarrolladores y usuarios juntos y ayuda a superar muchos de los problemas de comunicación y comportamiento humano en el proceso de obtención de requisitos. Se apoya en la dinámica de grupos, en el uso de ayudas visuales en las reuniones a fin de mejorar la comunicación y comprensión, en el mantenimiento de un proceso organizado en el que los papeles de cada participante están perfectamente definidos, y en la filosofía de la documentación.

Las características de una sesión de trabajo tipo JAD se pueden resumir en los siguientes puntos:

- Se establece un equipo de trabajo cuyos componentes y responsabilidades están perfectamente identificados. Los perfiles que se pueden distinguir son los siguientes:
 - Moderador, con amplios conocimientos de la metodología de trabajo, dinámica de grupos, psicología del comportamiento y de los procesos de la organización.
 - Promotor, persona que ha impulsado el desarrollo.
 - Jefe de proyecto, responsable de la implantación del proyecto.

- Especialista en modelización, responsable de la elaboración de los modelos en el transcurso de la sesión.
 - Desarrolladores, aseguran que los modelos son correctos y responden a los requisitos especificados.
 - Usuarios, responsables de definir los requisitos del sistema y validarlos.
- Se llevan a cabo pocas reuniones, de larga duración y muy bien preparadas.
 - Durante la propia sesión se elaboran los modelos empleando diagramas fáciles de entender y mantener, generalmente utilizando herramientas CASE.
 - Al finalizar la sesión se obtienen un conjunto de modelos que deberán ser aprobados por los participantes.

Para llevar a cabo una sesión JAD, es necesario realizar una serie de actividades antes de su inicio, durante el desarrollo y después de su finalización.

- Antes del inicio, se define el ámbito y la estructura del proyecto, los productos a obtener, se prepara el material necesario para la sesión, se determina el lugar donde se van a llevar a cabo, se seleccionan los participantes y se sugiere una agenda de trabajo.
- Durante el desarrollo se identifican las salidas del proyecto y se debe conseguir el consenso entre los participantes de modo que se materialice en los modelos.
- Después de la finalización se valida la información de la sesión y se generan los productos de la metodología de trabajo propuesta.

En las sesiones de trabajo tipo JAD se distinguen dos tipos de productos: los de preparación (historia y contexto del proyecto, objetivos y límites, actividades del entorno del negocio que pueden afectar al éxito del proyecto y los beneficios) y los de resultado de las sesiones de trabajo, que se establecen con anterioridad al inicio de las reuniones.

3.1.2. El prototipado.

El prototipado constituye un medio a través del cual se simula el aspecto visual del sistema mediante la representación de los conceptos, componentes, objetos gráficos, entradas y salidas requeridas para la ejecución de cada función en respuesta a las necesidades planteadas. Su objetivo es elaborar un modelo o maqueta de las interfaces entre el sistema y el usuario (formatos de pantallas, informes, formularios, etc.), que ayude a éste a comprender cómo se producirá la interacción con el sistema.

Con el prototipado se crea, mediante la implementación de sólo algunos requisitos del software, un sistema que ilustra los rasgos más sobresalientes, a fin de que pueda ser examinado por los usuarios para aprender sobre las necesidades reales que tienen. Por tanto, es importante que el usuario colabore en el desarrollo del prototipo o de la maqueta ¹ y evalúe hasta qué punto las funciones se implementan de forma apropiada y cubren los requisitos identificados.

¹ El prototipo es algo que demuestra el comportamiento de una parte del sistema deseado. La maqueta es algo que demuestra la apariencia del sistema deseado.

La aplicación de la técnica del prototipado consta de los siguientes pasos:

- Estudio preliminar de los requisitos del usuario.
- Proceso iterativo consistente en:
 - Construir un prototipo.
 - Evaluarlo con los usuarios.
 - Formular nuevos requisitos.
 - Desechar el prototipo.

Con objeto de conseguir una interfaz eficiente y compatible con las necesidades de los usuarios los aspectos clave en el diseño de prototipos son: la identificación de los usuarios a los que va dirigido, teniendo en cuenta que debe responder a diferentes individualidades, con distintos conocimientos y habilidades, qué funciones tienen asignadas, y qué tipo de información requerirán para llevar a cabo dichas funciones.

Asimismo, la forma en que se establezca la comunicación entre el usuario y el sistema será decisiva para conseguir su aceptación. Por tanto, al iniciar el diseño de prototipos de pantallas, es imprescindible tener en cuenta las siguientes consideraciones:

- Utilizar conceptos, términos y símbolos familiares al usuario, de modo que sea fácil de aprender y comprender.
- Mantener la coherencia dentro del propio sistema y entre sistemas proporcionando abreviaturas estándar, aplicando las mismas reglas de interacción a través de toda la interfaz, y utilizando el mismo formato para los mensajes de error y comandos con significado similar.
- Facilitar la exploración del sistema sin riesgo, permitiendo interrumpir y deshacer las acciones realizadas.
- Dificultar la selección de acciones destructivas y no reversibles, pidiéndole verificación de cualquier acción que conlleve un riesgo importante.
- Proporcionar información sobre el estado de ejecución de las funciones.
- Agrupar las funciones de forma lógica y presentar primero las más utilizadas.
- Buscar la eficiencia en el diálogo evitando cambios frecuentes entre los dispositivos de entrada, tales como el ratón y el teclado.

Una vez considerados estos aspectos, para definir los formatos individuales de las pantallas se realiza un análisis de la información a presentar en cada una de ellas, el cual se centrará en los siguientes puntos:

- Identificar los diferentes tipos de información con el fin de organizar la pantalla en áreas específicas.

- Estudiar el espacio disponible en las pantallas para determinar qué datos y en qué situación deben aparecer en las mismas.
- Intentar agrupar los datos relacionados y mostrar sólo los esenciales para la ejecución de la función o para la toma de una decisión, eliminando todas las entradas innecesarias.
- Mantener la coherencia entre la entrada y la visualización de datos.
- Proteger al usuario frente a acciones que podrían provocar errores, desactivando los comandos que no son operativos en ese contexto.

Finalmente, con objeto de atraer la atención del usuario y mejorar su interacción con el sistema, se deben prestar especial atención a aquellos atributos relacionados con el aspecto estético como son el color y el sonido.

3.1.3. El Análisis de los Factores Críticos de Éxito.

Esta técnica proporciona un proceso de educación sistemático, consistente en identificar y concentrarse en un pequeño conjunto de factores críticos de los que depende el éxito y efectividad del sistema. Aunque la identificación de los factores críticos correctos es, a veces, una labor compleja, esta técnica es bastante útil cuando el sistema es técnicamente complejo.

3.1.4. Las Entrevistas.

Ésta es, quizás, la técnica más importante y la más utilizada en la identificación de requisitos, bien de por sí, o bien como parte integrante de las técnicas de educación de alto nivel, para obtener la información detallada de un individuo. La entrevista es una técnica estructurada que precisa habilidades sociales, capacidad de escuchar y conocimiento de diversas tácticas de entrevista.

Los pasos que configuran la entrevista son cuatro:

1. La identificación de los usuarios y personas interesadas en el sistema, a entrevistar.
2. La preparación de los guiones de la entrevista.
3. La realización de la entrevista.
4. La consolidación de la entrevista.

Una vez identificados los usuarios a entrevistar, es imprescindible enviarles un guión previo sobre los puntos a tratar. Dicho guión ha de ser lo suficientemente detallado como para cubrir todos los puntos de interés y, en su caso, dar oportunidad al usuario de preparar la documentación que se precise sobre el sistema, pero no excesivamente extenso y con excesivo detalle, pues esto puede provocar el rechazo del usuario.

El guión de la entrevista se elaborará atendiendo al perfil y a las responsabilidades del usuario a entrevistar. Evidentemente, no es lo mismo una entrevista a un responsable o director de área, donde lo que se busca es la determinación de los aspectos funcionales generales, que una entrevista a los restantes usuarios, cuyo objeto es detallar aspectos funcionales más concretos.

En cualquier caso, hay cierto tipo de preguntas en las entrevistas, por ejemplo, grado de satisfacción con el sistema actual, etc., cuya respuesta puede ser evaluada atendiendo al siguiente baremo:

1 = Es costoso o difícil trabajar con él.

2 = Es tolerable.

3 = Es mediano.

4 = Es satisfactorio, pero podría mejorarse.

5 = Es excelente.

En la realización de la entrevista ésta se podrá estructurar de acuerdo a tres patrones diferentes:

- a) Estructura de pirámide. El entrevistador comienza con preguntas muy detalladas y frecuentemente cerradas, es decir, preguntas que limitan las respuestas disponibles al entrevistado, para luego expandir el tema hacia aspectos más generales mediante preguntas abiertas, es decir, preguntas en las que la respuesta del entrevistado no está limitada.
- b) Estructura de embudo. El entrevistador comienza con preguntas generales y abiertas para terminar con preguntas muy específicas y detalladas. Al seguir la entrevista un enfoque deductivo, esta manera de estructurarla suele ser la más utilizada.
- c) Estructura de rombo. Es una combinación de las dos anteriores. Se comienza con cuestiones específicas, luego se examinan cuestiones generales y se concluye con preguntas muy detalladas, de nuevo.

Asimismo, durante la realización de la entrevista se deberán comprobar periódicamente los errores de comunicación entre entrevistado y entrevistador, tales como: errores de interpretación de palabras, errores de enfoque, errores de recuerdo, ambigüedades del lenguaje natural, etc.

Una vez realizada la entrevista es necesario depurar y consolidar los resultados de la misma para asegurar que se han comprendido bien las especificaciones y requisitos del usuario. Para ello:

- Se producirá un resumen escrito de la entrevista, reorganizando las ideas y consolidando la información relacionada.
- Se utilizarán técnicas de representación libre, que ilustren el funcionamiento del sistema.
- Se especificarán los requisitos de forma sencilla y medible, de tal manera que se puedan asignar prioridades a los mismos.
- Finalmente, se validará todo conjuntamente con el usuario.

4. ANÁLISIS Y NEGOCIACIÓN DE LOS REQUISITOS IDENTIFICADOS.

Una vez identificados y reunidos los requisitos, se necesita evaluar o analizar los mismos a fin de determinar que son adecuados para el cliente o usuario. Esto es, que los requerimientos que ha de satisfacer el sistema software:

- Tienen un nivel aceptable de riesgo, tanto desde la perspectiva de la viabilidad técnica, como del coste.
- Son completos, es decir, están todos.
- Son correctos, o sea, no tienen errores.
- No son ambiguos, es decir, no tienen una doble interpretación y son, por tanto, claros, concretos y precisos.

El análisis y negociación de los requisitos engloba el aprendizaje sobre el problema a resolver y la comprensión de las necesidades de los usuarios y de todas las restricciones que la solución debe cumplir; y, por consiguiente, puede verse como la definición del rango de todas las posibles soluciones software a un problema, que cumplen todas las restricciones impuestas. Las fuentes de restricciones pueden ser los usuarios, los clientes, el equipo de desarrollo, la propia tecnología y las leyes y estándares.

Durante esta etapa de la Ingeniería de Requisitos se investigan todos los aspectos del dominio del problema. A estos efectos, los elementos que se suelen considerar, y que son los mismos que luego se van a utilizar para describir el comportamiento externo del software, son: los objetos, las funciones y los estados, los cuales se pueden describir en múltiples niveles de detalle:

- Un objeto es una entidad del mundo real claramente delimitada, perteneciente al dominio del problema. Por ejemplo, en un sistema que debe mostrar el estado de cuentas de los clientes, tanto «cuenta» como «cliente» son objetos en tal sistema.
- Una función es una tarea o actividad que tiene que ser ejecutada por el sistema que se está especificando, para resolver el problema. En el ejemplo anterior, una función será imprimir las cuentas de un determinado cliente.
- Un estado es una condición del sistema, objeto o función, que ayuda a capturar su historia, de forma que define cómo se comporta en condiciones específicas.

Asimismo, durante esta etapa se clasificarán los requisitos atendiendo a diversos criterios, tales como:

- Por prioridades: obligatorios, deseables y no esenciales.
- Funcionales y no funcionales.
- Por coste de implementación.
- Si son requisitos sobre el proceso o sobre el producto.

Por último, dado que en el proceso de la Ingeniería de Requisitos intervienen distintos individuos, a veces con intereses enfrentados, se negociarán los posibles conflictos que puedan surgir acerca de los requisitos. Los conflictos no deben rechazarse y resolverse por decreto, sino que deben negociarse pues muchas veces son fuente de nuevos requisitos.

4.1. TÉCNICAS DE ANÁLISIS.

Aunque las Técnicas de Análisis serán objeto de estudio en otros temas, se adelantarán aquí, no obstante, algunas ideas generales al respecto.

Para llevar a cabo las tareas asociadas al análisis de los requisitos del software identificados, se han desarrollado una serie de técnicas cuya finalidad principal es facilitar al analista la aplicación sistemática de los principios fundamentales del análisis.

Las técnicas ayudan a organizar la información reunida durante el análisis, relacionando las perspectivas de diferentes personas y sacando a la luz y resolviendo posibles conflictos, y evitando el diseño interno del software.

Un aspecto muy importante a tener en cuenta en el análisis es evitar descender al diseño del software. El análisis tiene como objeto comprender el problema y debe reflejar las necesidades, mientras que el diseño debe reflejar cómo resolverlas.

El objetivo de los métodos y técnicas desarrollados para la realización del análisis de los requisitos del software es suministrar una serie de mecanismos que permitan representar el dominio de la información, de tal forma que de dicha representación se derive la función del software.

En general, todos los métodos y técnicas de análisis tienen las siguientes características principales:

- Proporcionan un medio para definir los límites del sistema.
- Proporcionan mecanismos para el análisis del dominio de la información.
- Proporcionan métodos de representación funcional.
- Proporcionan un medio para definir particiones, abstracciones y proyecciones.
- Permiten representar la visión lógica y física del sistema.

El análisis permite realizar la descripción de un problema, y, tal como se indicó, los elementos básicos que se utilizan para ello son los objetos, las funciones y los estados. Desde este punto de vista, un problema puede ser modelizado atendiendo a tres perspectivas: estática, dinámica y funcional. Las tres son complementarias y juntas proporcionan la posibilidad de construir un modelo del problema.

- En la descripción estática del problema (enfoque orientado a datos y enfoque orientado a objetos), se identifica la estructura estática de los objetos y sus relaciones, obteniéndose una imagen estática del dominio del problema.
- En la descripción dinámica del problema (enfoque orientado a estados), se describen los aspectos del sistema que cambian con el tiempo y que, juntos, caracterizan los distintos estados que puede presentar un problema, así como las condiciones y eventos que permiten transitar entre estos estados.
- En la descripción funcional del problema (enfoque orientado a procesos), se describe cómo se transforman los datos en el contexto del problema.

En base a los elementos básicos del dominio del problema, se muestra a continuación una clasificación de las distintas técnicas de análisis. Todas ellas se estudiarán en otros temas.

PRINCIPALES TÉCNICAS DE ANÁLISIS

TÉCNICAS DE ANÁLISIS ORIENTADAS A OBJETOS	TÉCNICAS DE ANÁLISIS ORIENTADAS A FUNCIONES	TÉCNICAS DE ANÁLISIS ORIENTADAS A ESTADOS
<ul style="list-style-type: none"> • Modelización Entidad-Relación • Modelización de objetos 	<ul style="list-style-type: none"> • Diagramas de flujo de datos • Modelo funcional • Definición de requisitos estructurados • Técnica de análisis y diseño estructurado • Análisis estructurado y especificación del sistema 	<ul style="list-style-type: none"> • Diagramas de estado

5. LA DOCUMENTACIÓN DE LOS REQUISITOS.

La fase de Requisitos del Software concluye una vez que se tiene una descripción completa del comportamiento externo del software y dicha descripción se escribe en el documento denominado «Documento de Especificación de Requisitos del Software» (ERS).

Los objetivos que pretende el «Documento de Especificación de Requisitos del Software» (ERS, en adelante) son los siguientes:

- Proporcionar los medios de comunicación entre todas las partes implicadas en el sistema: clientes, usuarios, analistas y diseñadores.
- Servir como base para las actividades de prueba y verificación.
- Ayudar al control de la evolución del sistema software.

El documento ERS debe incluir una descripción completa y concisa de toda la interfaz externa del sistema con su entorno, incluido el resto del software, puertos de comunicación, hardware y usuarios. Es decir, debe incluir tanto los requisitos de comportamiento del sistema (funcionales), que son aquellos que definen lo que hace éste y la información que maneja, como los requisitos que no son de comportamiento, esto es, aquellos que definen los atributos del sistema según realiza su trabajo (eficiencia, fiabilidad, seguridad, etc.).

Por contra, un documento ERS no debe incluir los requisitos del proyecto (planificaciones, hitos, etc.), ni el diseño, ni los planes de control del producto (gestión de configuración, garantía de calidad, etc.).

Un documento ERS debe reunir las siguientes características:

- Correcto. Cada requisito establecido debe representar algo requerido para el sistema.
- No Ambiguo. Cada requisito establecido tiene una sola interpretación.
- Completo. Debe incluir todo lo que el software tiene que hacer.
- Verificable. Se ha de poder comprobar, mediante un proceso efectivo y de coste limitado, que el producto reúne cada requisito establecido.
- Consistente. Cada requisito no puede estar en conflicto con otros requisitos.
- Modificable. La estructura y estilo del documento debe hacer fáciles los cambios.
- Conciso, comprensible por el usuario y organizado.
- Referenciado. Cada requisito debe estar calificado y debidamente referenciado.

El contenido del documento ERS se puede organizar de diversas maneras. Un estándar de organización muy utilizado comercialmente es el del IEEE/ANSI 830-1993. Según dicho estándar, el contenido del documento ERS se puede organizar como sigue:

1. INTRODUCCIÓN. (Visión de la ERS).

- 1.1 - Propósito del documento.
- 1.2 - Alcance del documento.
- 1.3 - Definiciones de términos dudosos, acrónimos y abreviaturas.
- 1.4 - Referencias.
- 1.5 - Estructura del documento.

2. DESCRIPCIÓN GENERAL. (Factores que afectan al producto software y al documento ERS).

- 2.1 - Perspectiva del producto.
- 2.2 - Funciones del producto.
- 2.3 - Características de usuario.
- 2.4 - Restricciones.
- 2.5 - Dependencias de los requisitos.
- 2.6 - Requisitos incorporables a futuras versiones.

3. REQUISITOS ESPECÍFICOS. (Especificación con suficiente detalle para diseñadores y probadores. Deben incluir entrada, salida y funciones).

- 3.1 - Interfaces externos.
- 3.2 - Funciones.
- 3.3 - Requisitos de rendimiento.
- 3.4 - Restricciones lógicas de base de datos y de diseño, y cumplimiento con estándares.
- 3.5 - Atributos del sistema software.

5.1. ESPECIFICACIÓN DE LOS REQUISITOS FUNCIONALES.

Para especificar correctamente los requerimientos funcionales que ha de cumplir el software y, consecuentemente, escribirlos o documentarlos de manera adecuada, existen diversas técnicas de especificación de requisitos cuyos objetivos se centran básicamente en:

- Reducir la ambigüedad propia del lenguaje natural.
- Permitir que el ordenador haga el trabajo difícil de detectar inconsistencias, incompletudes y ambigüedades.
- Servir de ayuda y ser comprensible a usuarios y clientes no especializados en ordenadores.
- Servir de base para el diseño y pruebas del sistema.
- Proporcionar una base para la generación de prototipos y de pruebas automáticas del sistema.
- Ayudar a organizar la información en el documento ERS y facilitar la modificación del mismo.

La capacidad del escritor de requisitos para especificar correctamente un sistema depende del número de técnicas que conozca y cómo y cuándo aplicarlas. Las principales técnicas de especificación de requisitos funcionales o de comportamiento externo del sistema, organizadas por su orientación particular, se presentan en el siguiente cuadro.

TÉCNICAS ORIENTADAS A ESTADOS	TÉCNICAS ORIENTADAS A FUNCIONES	TÉCNICAS ORIENTADAS A OBJETOS	MÉTODOS FORMALES
<ul style="list-style-type: none">• Máquinas de estados finitos.• Diagramas de estados.• Sistema de validación de ingeniería de requisitos (REVS).• Redes de Petri.	<ul style="list-style-type: none">• Tablas y árboles de decisión.• Lenguaje de diseño de programas.	<ul style="list-style-type: none">• Análisis orientado a objetos.• Desarrollo de sistemas de Jackson.	<ul style="list-style-type: none">• Lenguaje Z.

5.2. ESPECIFICACIÓN DE LOS REQUISITOS NO FUNCIONALES.

Cuando hablamos de requisitos del software que no son de comportamiento o, si se quiere, funcionales, nos estamos refiriendo básicamente a cómo especificar el grado de calidad del software, como parte de los requisitos del sistema que se está construyendo.

En este sentido, los principales requisitos no funcionales del software (requisitos de calidad), tal como muestra la figura, son los siguientes:

- Portabilidad es el grado de facilidad por el que el software que se está ejecutando en la plataforma hardware para la que se destinó puede ser ejecutado en otro entorno de computadoras.

Cuantificar la portabilidad es realmente imposible. Una aproximación puede ser considerar el tiempo que se tardaría en llevar el sistema a una determinada plataforma; otra puede ser definir la portabilidad basándose en el lenguaje fuente, en el sistema operativo de la plataforma y en el compilador.

- Fiabilidad es la capacidad del software para comportarse consistentemente de una manera aceptable por el usuario, en el entorno para el que se destinó.

Respecto a la fiabilidad, el documento ERS necesita expresar que el software que se va a entregar funciona, esto es, carece de errores. Una forma de saber cuántos errores hay en el software que se entrega es a través de la Técnica de Monte Carlo, mediante el análisis estadístico de sucesos aleatorios.

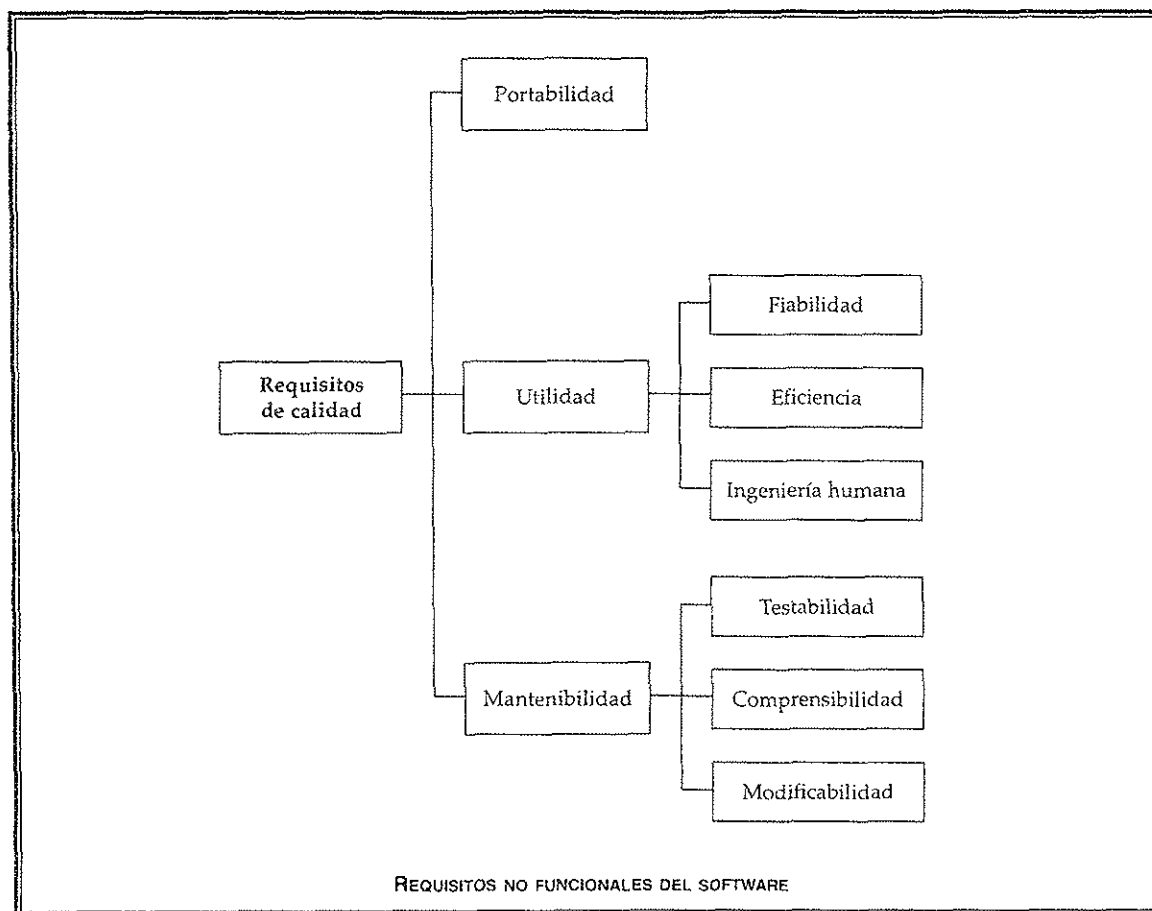
Antes de la fase de pruebas, se insertan secretamente un número de errores preseleccionados, que se añaden a los ya existentes en el sistema. El equipo de pruebas comienza su trabajo sin conocer cuáles ni cuántos errores se han introducido, y al acabar la fase de pruebas habrán descubierto un número de errores, algunos de los cuales serán los preseleccionados. Entonces, según esta técnica, se mantiene la siguiente proporción:

$$\frac{\text{Número de errores preseleccionados detectados}}{\text{Número total de errores detectados}} = \frac{\text{Número total de errores preseleccionados}}{\text{Número total de errores en el sistema}}$$

- La Eficiencia se refiere al nivel de recursos del sistema que usa el software, tales como el ciclo de máquina, la memoria, el espacio en disco, los canales de comunicación, etc.

La eficiencia se puede medir a través de la cuantificación de la capacidad, esto es, para cada posible entrada, cuántos terminales, usuarios, etc., se esperan; de la degradación del servicio y de las restricciones de tiempo, las cuales definen los requisitos de tiempo de respuesta para el software y su entorno.

- La Ingeniería Humana se refiere a la interface de usuario. Los dos aspectos fundamentales relacionados con la ingeniería humana son el tipo de estilo interactivo utilizado para la interfaz de usuario del sistema (lenguaje de comandos, sistema basado en menús, uso de iconos, ventanas, etc.) y el uso de mensajes de error. (No se debe olvidar que muchos sistemas se descartan debido a sus pobres interfaces de usuario).
- Por último, la testabilidad, la capacidad de comprensión y la modificabilidad están estrechamente relacionados y resulta muy difícil cuantificarlos, mientras que, sin embargo, repercuten muy fuertemente en el coste del ciclo de vida del producto. Algunas sugerencias son: especificar estándares de programación, sugerir lenguajes de más alto nivel, puesto que son más mantenibles y modificables, etc.



6. LA VISIÓN DE LOS REQUISITOS SEGÚN LA METODOLOGÍA MÉTRICA V.3.

La Metodología de Planificación, Desarrollo y Mantenimiento de Sistemas de Información Métrica versión 3, contempla lo que aquí hemos llamado «Fase o Ingeniería de Requisitos» en el proceso de Estudio de Viabilidad del Sistema y, sobre todo, en el proceso de Análisis del Sistema de Información.

En el Estudio de Viabilidad del Sistema (EVS) la actividad EVS 3 «Definición de requisitos del sistema» incluye la determinación de los requisitos generales mediante una serie de sesiones de trabajo con los usuarios participantes, que hay que planificar y realizar. Una vez finalizadas, se analiza la información obtenida definiendo los requisitos y sus prioridades, y se incorporan al catálogo de requisitos que servirá para el estudio y valoración de las distintas alternativas de solución que se propongan.

Las tareas que contempla esta actividad son las siguientes:

TAREA		PRODUCTOS	TÉCNICAS Y PRÁCTICAS	PARTICIPANTES
EVS 3.1	Identificación de las Directrices Técnicas y de Gestión.	• Catálogo de Normas.	• Catalogación.	• Jefe de Proyecto. • Analistas. • Usuarios Expertos.
EVS 3.2	Identificación de Requisitos.	• Identificación de Requisitos.	• Sesiones de Trabajo.	• Jefe de Proyecto. • Analistas. • Usuarios Expertos.
EVS 3.3	Catalogación de Requisitos.	• Catálogo de Requisitos.	• Catalogación.	• Jefe de Proyecto. • Analistas. • Usuarios Expertos.

En el proceso de Análisis del Sistema de Información (ASI), dado que su objetivo es obtener una especificación detallada del sistema de información que satisfaga las necesidades de información de los usuarios y sirva de base para el posterior diseño del sistema, cobra una importancia vital la actividad ASI 2 «Establecimiento de Requisitos».

En esta actividad se lleva a cabo la definición, análisis y validación de los requisitos a partir de la información facilitada por el usuario, completándose el catálogo de requisitos obtenido en la Definición del Sistema. El objetivo de esta actividad es obtener un catálogo detallado de los requisitos, a partir del cual se pueda comprobar que los productos generados en las actividades de modelización se ajustan a los requisitos de usuario.

Esta actividad se descompone en las tareas que se indican a continuación, que, si bien tienen un orden, exigen continuas realimentaciones y solapamientos, entre sí y con otras tareas realizadas en paralelo. No es necesaria la finalización de una tarea para el comienzo de la siguiente. Lo que se tiene en un momento determinado es un catálogo de requisitos especificado en función de la progresión del proceso de análisis.

TAREA		PRODUCTOS	TÉCNICAS Y PRÁCTICAS	PARTICIPANTES
ASI 2.1	Obtención de Requisitos.	• Catálogo de Requisitos. • Modelo de Casos de Uso.	• Sesiones de Trabajo. • Catalogación. • Casos de Uso.	• Usuarios Expertos. • Analistas.
ASI 2.2	Especificación de Casos de Uso.	• Catálogo de Requisitos. • Modelo de Casos de Uso. • Especificación de Casos de Uso.	• Sesiones de Trabajo. • Catalogación. • Casos de Uso.	• Usuarios Expertos. • Analistas.

.../...

.../...

ASI 2.3	Análisis de Requisitos.	<ul style="list-style-type: none">• Catálogo de Requisitos.• Modelo de Casos de Uso.• Especificación de Casos de Uso.	<ul style="list-style-type: none">• Sesiones de Trabajo.• Catalogación.• Casos de Uso.	<ul style="list-style-type: none">• Usuarios Expertos.• Analistas.
ASI 2.4	Validación de Requisitos.	<ul style="list-style-type: none">• Catálogo de Requisitos.• Modelo de Casos de Uso.• Especificación de Casos de Uso.	<ul style="list-style-type: none">• Sesiones de Trabajo.• Catalogación.• Casos de Uso.	<ul style="list-style-type: none">• Usuarios Expertos.• Analistas.

6.1. OBTENCIÓN DE REQUISITOS.

En esta tarea comienza la obtención detallada de información mediante sesiones de trabajo con los usuarios. Se recoge información de los requisitos que debe cumplir el software, teniendo en cuenta las posibles restricciones del entorno, tanto hardware como software, que puedan afectar al sistema de información. Asimismo, también se definen las prioridades que hay que asignar a los requisitos, considerando los criterios de los usuarios acerca de las funcionalidades a cubrir.

Los principales tipos de requisitos que se deben especificar son los funcionales, los de rendimiento, los de seguridad, los de implantación y los de disponibilidad del sistema. En el caso de orientación a objetos se especifican, además, los casos de uso asociados a los requisitos funcionales.

6.2. ESPECIFICACIÓN DE CASOS DE USO.

Esta tarea es obligatoria en el caso de orientación a objetos, y opcional en el caso de análisis estructurado, como apoyo a la obtención de requisitos. Su objetivo es especificar cada caso de uso identificado, desarrollando el escenario.

Para completar los casos de uso, es preciso especificar información relativa la descripción del escenario, es decir, cómo un actor interactúa con el sistema, y cuál es la respuesta obtenida; a las precondiciones y poscondiciones; a la identificación de interfaces de usuario; y a las condiciones de fallo que afectan al escenario, así como la respuesta del sistema.

6.3. ANÁLISIS DE REQUISITOS.

En esta tarea se estudia la información para detectar inconsistencias, ambigüedades, duplicidad o escasez de información. También se analizan las prioridades establecidas por el usuario y se asocian los requisitos relacionados entre sí. El análisis de los requisitos y de los casos de uso asociados permite identificar funcionalidades o comportamientos comunes, reestructurando la información de los casos de uso a través de las generalizaciones y relaciones entre ellos.

6.4. VALIDACIÓN DE REQUISITOS.

Mediante esta tarea, los usuarios confirman que los requisitos especificados en el catálogo de requisitos, así como los casos de uso, son válidos, consistentes y completos.

Dentro del proceso de Análisis del Sistema de Información, la actividad ASI 9 «Análisis de Consistencia y Especificación de Requisitos» tiene por objeto garantizar la calidad de los distintos modelos generados en el Análisis y asegurar que los usuarios y los analistas tienen el mismo concepto del sistema. Para ello, se verifica la calidad técnica de cada modelo, se asegura la coherencia entre los distintos modelos y se valida el cumplimiento de los requisitos. En esta actividad también se elabora el documento ERS, como producto para la aprobación formal, por parte del usuario, de las especificaciones del sistema.

El objetivo de la tarea de «Análisis de Consistencia entre Modelos» es asegurar que los modelos son coherentes entre sí, comprobando la falta de ambigüedades o duplicación de información. Las diferentes comprobaciones varían en función del tipo de desarrollo, aunque, en general, son matices entre los elementos comunes de los distintos modelos.

Los análisis de consistencia propuestos en Desarrollo Estructurado son:

- Modelo Lógico de Datos Normalizado/Modelo de Procesos.
- Modelo Lógico de Datos Normalizado/Interfaz de Usuario.
- Modelo de Procesos/Interfaz de Usuario.

Los propuestos en Desarrollo Orientado a Objetos son:

- Modelo de Clases/Diagramas Dinámicos.
- Modelo de clases/Interfaz de usuario.
- Análisis de la Realización de los Casos de Uso/Interfaz de Usuario.

En la tarea de «Elaboración de la Especificación de Requisitos Software» se aborda la elaboración de la especificación de los requisitos de software, una vez validados los modelos. Este producto incorporará la información necesaria para la aprobación final del Análisis del Sistema de Información, según el siguiente índice:

- Introducción.
- Ámbito y alcance.
- Participantes.
- Requisitos del sistema de información.
- Visión general del sistema de información.
- Referencia de los productos a entregar.
- Plan de acción.

ACTIVIDAD DE ELABORACIÓN DE LA ESPECIFICACIÓN DE REQUISITOS SOFTWARE

TAREA	PRODUCTOS	TÉCNICAS	PARTICIPANTES
Verificación de los modelos	Especificac. Interfaz de Usuario Modelo lógico de datos Modelo de Procesos Modelo de Casos de Uso Especificación de Casos de Uso Modelo de Clases de Análisis Etcétera		<ul style="list-style-type: none"> • Analistas • Equipo de Arquitectura
Análisis de Consistencia entre modelos	Resultado de Análisis de Consistencia Especificac. Interfaz de Usuario Modelo lógico de datos Modelo de Procesos Modelo de Casos de Uso Especificación de Casos de Uso Modelo de Clases de Análisis Etcétera	<ul style="list-style-type: none"> • Matricial • Cálculo de accesos lógicos • Caminos de accesos lógicos en consultas 	<ul style="list-style-type: none"> • Analistas • Equipo de Arquitectura
Validación de los modelos	Especificac. Interfaz de Usuario Modelo lógico de datos Modelo de Procesos Modelo de Casos de Uso Especificación de Casos de Uso Modelo de Clases de Análisis Etcétera	<ul style="list-style-type: none"> • Prototipado 	<ul style="list-style-type: none"> • Analistas • Usuarios expertos
Elaboración de la Especificación de Requisitos Software (ERS)	Especificación de los Requisitos Software (ERS)		<ul style="list-style-type: none"> • Analistas

BIBLIOGRAFÍA

- Ingeniería del Software. Un enfoque práctico. ROGER S. PRESSMAN. Ed. McGraw Hill.
- Ingeniería del Software. IAN SOMMERVILLE. Ed. Addison-Wesley.
- Metodología de Planificación, Desarrollo y Mantenimiento de Sistemas de Información. Métrica versión 3. MINISTERIO PARA LAS ADMINISTRACIONES PÚBLICAS.
- Metodología de Planificación y Desarrollo de Sistemas de Información. Métrica versión 2.1. Guía de Técnicas. MINISTERIO PARA LAS ADMINISTRACIONES PÚBLICAS. Ed. Tecnos.
- Temario de las pruebas selectivas para ingreso en el Cuerpo Superior de Sistemas y Tecnologías de la Información de la Administración del Estado. ASTIC.
- Temario de las pruebas selectivas para el acceso, por promoción interna, al Cuerpo de Gestión de Sistemas e Informática de la Administración del Estado. MINISTERIO PARA LAS ADMINISTRACIONES PÚBLICAS.
- Temario del Máster en Ingeniería del Software. FACULTAD DE INFORMÁTICA. UNIVERSIDAD POLITÉCNICA DE MADRID. Ed. Centro de Estudios Financieros.
- Documentación y papeles de trabajo de Ingeniería de Sistemas de Información. RAMÓN ORTIGOSA.



