Tabla de Contenidos

Índice de figuras

1. Introducción

La calidad siempre ha sido especificada en términos de los requerimientos. Por ejemplo, instituciones como la ISO la definen como "la totalidad de partes y características de un producto o servicio que influyen en su habilidad de satisfacer necesidades declaradas o implícitas" [ISO 1993]. Otros, la definen como "… sencillamente el conjunto de cualesquiera atributos requeridos en el software" [Ould 1994].

No importa la definición que se tome, todas ellas relacionan la calidad con los requerimientos, pero sorprendentemente estándares como el ISO 9000, que enfatizan en la definición y estandarización de los procesos, hasta antes del año 2000 no decían nada acerca de las funciones para establecer los requerimientos del sistema[1]. Inclusive, el Modelo de Madurez de Capacidades (CMM) del Software Engineering Institute (SEI), simplemente menciona la necesidad de administrar los requerimientos y solo ofrece algunos lineamientos de lo que esto involucra [Sawyer 1998].

La etapa del desarrollo de sistemas[2] en la cual se analiza el problema y se especifican las necesidades, se conoce como análisis de sistemas. Sin embargo, la labor relacionada con los requerimientos no debe terminar en esta etapa, sino que debe ir más allá, a través de todo el ciclo de vida, administrando los cambios que surgen en el transcurso del desarrollo.

Para denotar esta función dentro del desarrollo de sistemas, se inventó un término relativamente nuevo, *Ingeniería de Requerimientos*, que cubre todas las actividades involucradas en el descubrimiento, validación, documentación y mantenimiento de un conjunto de requerimientos de un sistema de computación. El uso del término "ingeniería" implica el uso de técnicas sistemáticas y repetitivas para asegurar que estos cumplan con características específicas, como lo son: completo, consistente, relevante, etc. [Sawyer 1998].

La etapa inicial de todo método de desarrollo involucra el análisis del problema y un producto final llamado *documento de requerimientos*[3]. Sobre éste se enfoca el presente trabajo, el cual pretende ofrecer al *ingeniero de requerimientos* lineamientos para poder evaluar la *calidad de los requerimientos*, más específicamente del documento que los respalda, para así ayudar a la calidad del producto final.

[1] La norma ISO 9000:2000 sí habla de requerimientos y de mejoramiento continuo. No se ha publicado una norma específica para interpretar la norma en el ámbito del desarrollo y mantenimiento de software (que sí existe para la edición anterior).

[2] Por lo general, cuando se habla de sistemas, se involucra tanto componentes de software y hardware. En este trabajo sistemas se referirá a solo la parte de software o sistemas de información, a menos que explícitamente se especifique de otro modo.

[3] El término *documento de requerimientos*, se utiliza para generalizar todos los términos utilizados por distintos autores y métodos de desarrollo. Inclusive, aplica cuando son almacenados en bases de datos y no propiamente en documentos.

2. Requerimientos

El punto central de este trabajo son los requerimientos, por lo que es conveniente definir primero qué son. Según el IEEE, un requerimiento es uno de los siguientes [Graham 1998]:

1. Una condición o capacidad necesitada por un usuario para resolver un problema o lograr un objetivo.

2. Una condición o capacidad que debe ser cumplida o poseída por un sistema o componente de sistema para satisfacer un contrato, estándar, especificación o cualquier otro documento formal impuesto.

3. Una representación documentada de una condición o capacidad como en 1 o 2.

Al decir que un requerimiento es una condición o capacidad, está dando a entender qué comprende un requerimiento, sin embargo, se requiere de una delimitación más exacta para describir qué son. Entonces, un requerimiento se puede clasificar de acuerdo con si es [Sawyer 1998]:

1. Una facilidad de usuario (ej. 'El sistema de contabilidad debe generar la información del estado de resultados al procesador de palabras XXX.').

2. Una propiedad general del sistema (ej. 'El sistema debe asegurar que la información personal nunca esté disponible sin autorización').

3. Una restricción específica en el sistema (ej. 'Los reportes debe imprimirse en menos de 15 segundos').

4. Una restricción en el desarrollo del sistema (ej. 'El sistema debe desarrollarse utilizando Ada').

Una clasificación más general y común, es agrupar a los requerimientos de acuerdo con si son funcionales o no. Los requerimientos enfocados a describir qué debe hacer un sistema, se denominan requerimientos *funcionales* y los requerimientos que describen cómo deben implementarse estos requerimientos se llaman *no funcionales*. Por ejemplo, un requerimiento funcional puede establecer que el sistema debe proveer de alguna facilidad de autenticación para identificar a los usuarios, por otro lado, un requerimiento no funcional puede establecer que el proceso de autenticación no puede durar más de 4 segundos. Sin embargo, en la práctica hacer la diferencia entre requerimientos *funcionales* y *no funcionales* no es tan sencillo [Sawyer 1998].

En general, los requerimientos describen *qué* debe hacer un sistema y no el *cómo* [Sawyer 1998]. Esto en teoría sería suficiente para distinguir qué es un requerimiento y es lo que debería contener el *documento de requerimientos*[4], sin embargo, en la práctica no es tan sencillo porque surge la paradoja del "qué versus el cómo" [Davis 1993].

Esta paradoja se puede resumir en "el cómo de una persona es el qué de otra". En otras palabras se pueden definir varios niveles donde es necesario especificar el qué. La Figura 2-1 muestra estos niveles en el software.

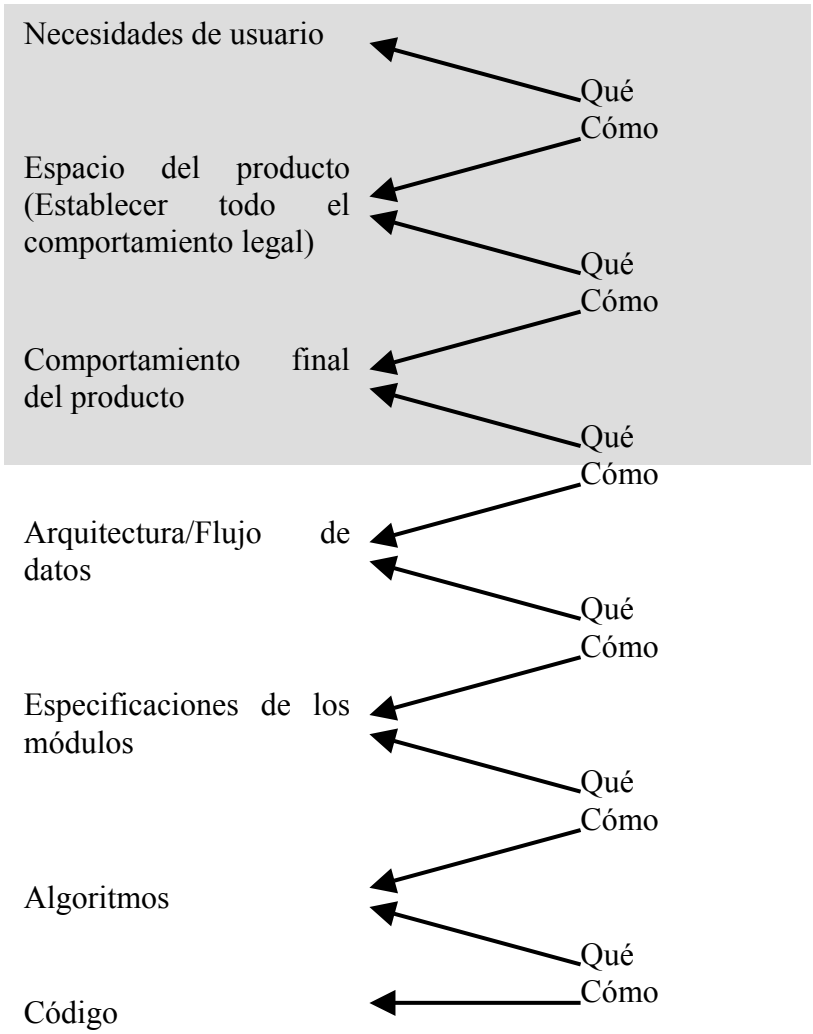


Figura 2-1 Paradoja del "qué versus el cómo" en los sistemas [Davis 1993].

[4] [Davis 1993] se refiere al documento de requerimientos como "Software Requirements Specification" o SRS, pero para estandarizar el uso de términos en este trabajo, se llamará "documento de requerimientos".

El primer nivel corresponde a la definición de las necesidades de los usuarios, donde claramente se tiene que especificar lo que el sistema debe hacer sin ninguna indicación de cómo llevarlo a cabo. Definir todo el conjunto de los posibles sistemas que pueden satisfacer los requerimientos del nivel uno, corresponde al *qué* del nivel dos, aunque esto sea el *cómo* del primer nivel.

Otro nivel sería especificar el comportamiento del producto final, que correspondería al *cómo* del nivel 2, pero es el *qué* de este tercer nivel. Igual sucede con el cuarto nivel, en el cual los requerimientos especificarían la arquitectura del sistema y el flujo de información. Esto diría cuáles son los componentes de sistema, sin especificar cómo se comportan internamente.

Se podría seguir describiendo los "qués" y "cómos" de cada grupo hasta llegar inclusive al código de máquina. Lo importante de esto es notar que no es tan sencillo distinguir el "qué" del "cómo".

Para efectos de este trabajo y lo que comprende la mayoría de los documentos de requerimientos en sistemas de información, siempre se estará trabajando sobre los primeros tres niveles. En otras palabras, el documento de requerimientos no debe entenderse como un documento de diseño, aunque esto no impide que el documento contenga aspectos de diseño, ya que no existe un límite claro entre especificación y diseño [Sawyer 1998]

Como se menciona en la definición del IEEE, estos requerimientos son originados por diversas fuentes: un usuario, un estándar, etc. Sin embargo, estos no son los que necesariamente crean la representación documentada de estos. En todo método de desarrollo, existe una etapa inicial cuya función principal es definirlos. Esta tiene como producto final el *documento de requerimientos*, el cual debería contener los requerimientos del futuro sistema.

3. La importancia de los requerimientos

El cliente o el proveedor, no importa quién, buscan siempre un producto de calidad para satisfacer sus necesidades. El primero para que el producto responda de la mejor manera a sus exigencias y el segundo para tener éxito en el negocio planteado.

Precisamente, muchos de los fracasos en los proyectos informáticos son consecuencia de fallas relacionadas con los requerimientos en cualquier punto del desarrollo. Por ejemplo, la mala planificación y control de un proyecto se debe, en gran parte, a la informalidad en la definición y especificación de los requerimientos, los cuales son necesarios para dimensionar el sistema [Rakos 1990].

Específicamente se señalan los siguientes puntos como causas del malogro de los proyectos informáticos[5] [Rakos 1990]:

a) **Fallas en el comienzo**: Los encargados del proyecto empiezan a programar sin saber por qué fue concebido el sistema, sin una descripción clara de lo que debe realizar y mucho menos se cuenta con un plan. La idea del usuario respecto del sistema por lo general es distinta a como los programadores se imaginan la solución.

b) **Fallas en las etapas de desarrollo**: En las etapas del desarrollo se pueden generar fallas al no tener bien documentados los resultados de etapas anteriores como el análisis y el diseño. Ocurre una mala interpretación entre las etapas.

c) **Fallas al final**: Debido a los límites en los tiempos y al agotado presupuesto, el proyecto se trata de acelerar sacrificando requerimientos sin estudiar el impacto real que tiene la eliminación de estos sobre el funcionamiento del nuevo sistema.

Aunque no se mencione directamente a los requerimientos como la causa del fracaso, se puede intuir que estos son la fuente de los problemas, no solamente por su ausencia, sino también por una mala administración de estos.

Adicional a esto, existen factores que por esta mala administración aumentan el riesgo de un proyecto informático. Algunos de los riesgos asociados a los requerimientos son los siguientes [McConnell 1996]:

a) A pesar que los requerimientos están establecidos, cambian.

[5] La referencia, [Rakos 1990], se enfoca solo sobre proyectos informáticos medianos y pequeños, sin embargo los mismos problemas surgen en proyectos grandes.

b) Los requerimientos están pobremente definidos y una definición posterior expande el alcance del proyecto.

c) Requerimientos adicionales son agregados posterior a la definición inicial.

d) Las especificaciones vagas de un área del producto consumen más tiempo del esperado.

La etapa en la cual se definen los requerimientos es fundamental para el desarrollo de sistemas, no solo desde el punto de vista de planificación y control de calidad (validación que el producto hace lo que debe hacer), si no también por el impacto que un error en los requerimientos tiene en el sistema y en los costos de éste.

Arreglar un requerimiento malo incrementa el costo según qué tan avanzada sea la etapa en la cual se detecte el error. Como muestra la Figura 3-1, el esfuerzo por etapa para arreglar algún error es muy superior en las etapas finales del proyecto que en las iniciales. Así por ejemplo, existe una relación de 200 a 1 entre reparar un error en la etapa de mantenimiento y la etapa de requerimientos [Davis 1993][6].

Etapa	Costo relativo de reparación
Requerimientos	0.1 – 0.2
Diseño	0.5
Codificación	1
Pruebas de Unidad	2
Pruebas de Aceptación	5
Mantenimiento	20

Figura 3-1. Costo (esfuerzo) para arreglar software en relación con la etapa de ciclo de vida [Davis 1993]

La explicación más aceptada de esta relación es porque el costo adicional de reparación puede involucrar no solamente el origen de éste, sino la inversión realizadas en el error en las etapas posteriores.

Precisamente esto es lo que se muestra en la Figura 3-2, en la cual se explica el problema.

[6] El costo de reparar defectos puede ser aún más dramático: Hewlett-Packard ha documentado relaciones de 1,000:1 entre las pruebas de aceptación y los requerimientos, y de 10,000:1 cuando el software ya ha sido distribuido entre sus clientes. El tipo de software referido incluye: sistemas operativos, sistemas administradores de bases de datos, sistemas de comunicación, controladores de dispositivos médicos, etc.

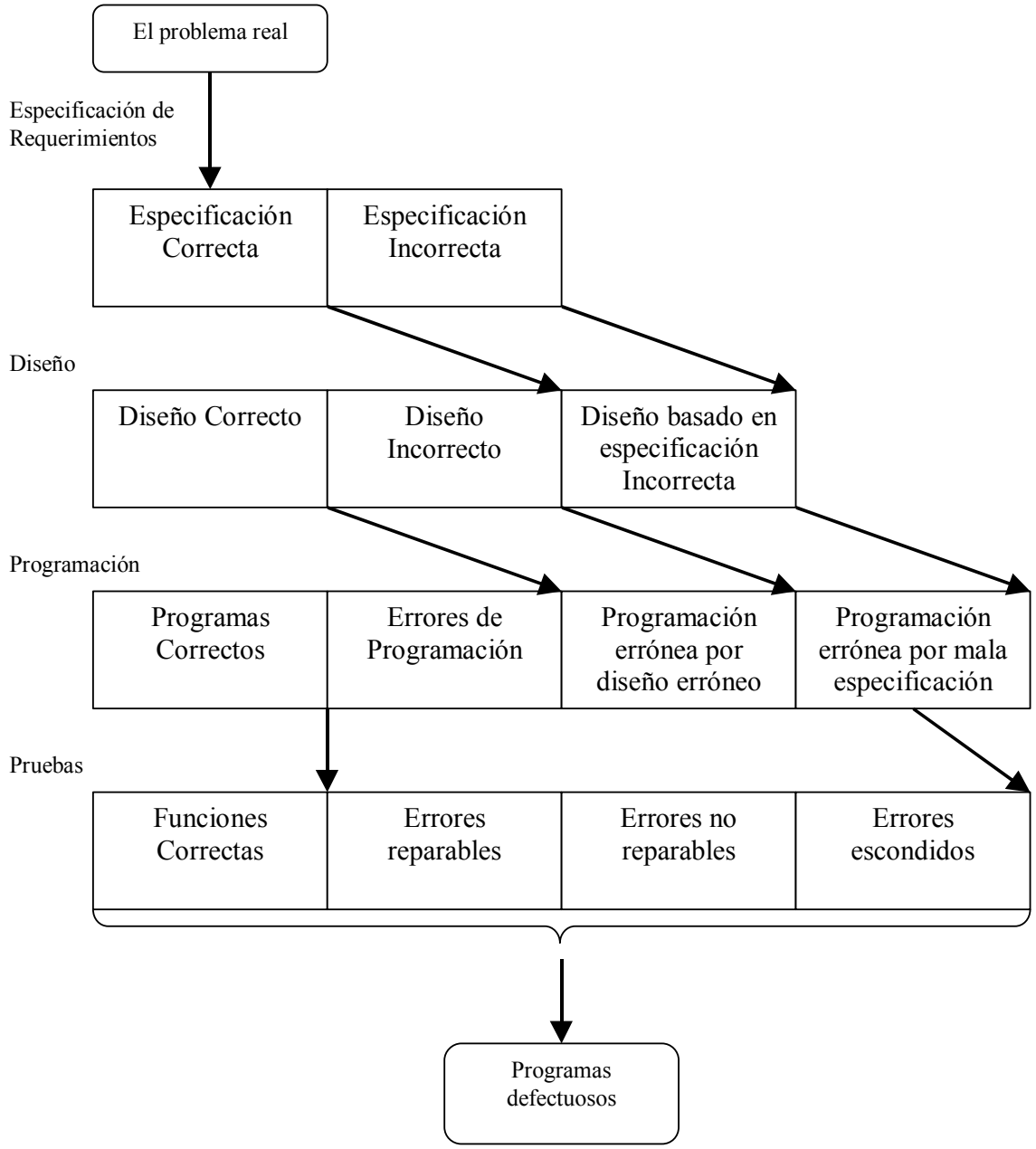


Figura 3-2. Efectos acumulativos de los errores [Davis 1993].

Suponiendo que se empieza con un problema real y se realiza la especificación de los requerimientos, una parte de ésta será correcta y la otra errónea. Luego se avanza a la etapa de diseño, donde se modela el sistema según la especificación utilizando tanto la parte buena como la errónea. El diseño basado en la parte correcta, generará también diseño correcto e incorrecto.

Después se avanza a la implementación, donde se extiende el problema. En esta etapa se realizarán programas incorrectos por motivo de la especificación incorrecta, por diseño incorrecto y errores propios de la programación. Cuando se llega a la etapa de pruebas, la parte de programación correcta se esperaría que trabaje correctamente. Algunos errores serán detectados y corregidos, otros serán solamente detectados y otros no serán detectados del todo [Davis 1993]. Cabe notar que las pruebas también son una fuente de falibilidad: pueden estar mal planteadas, ser incompletas o asistemáticas.

Aunque la mayoría de los desarrolladores de sistemas conocen la importancia de los requerimientos e intuyen los problemas relacionados con una mala definición y administración de los requerimientos, muchos sacrifican esta etapa, frecuentemente sucumbiendo a las presiones de usuarios, administradores y clientes.

La codificación prematura es causa de programas y diseños malos. Por lo general se tiende a ir directo a la programación, porque es lo que la mayoría hace mejor y usualmente existe una apatía a empezar trabajos difíciles como el análisis y el diseño [Rakos 1990].

De hecho, DeMarco reporta que el 56% de los errores detectados pueden ser rastreados hasta errores realizados en la etapa de requerimientos [Davis 1993]. Esto en parte se debe a la falta de atención en la etapa de análisis.

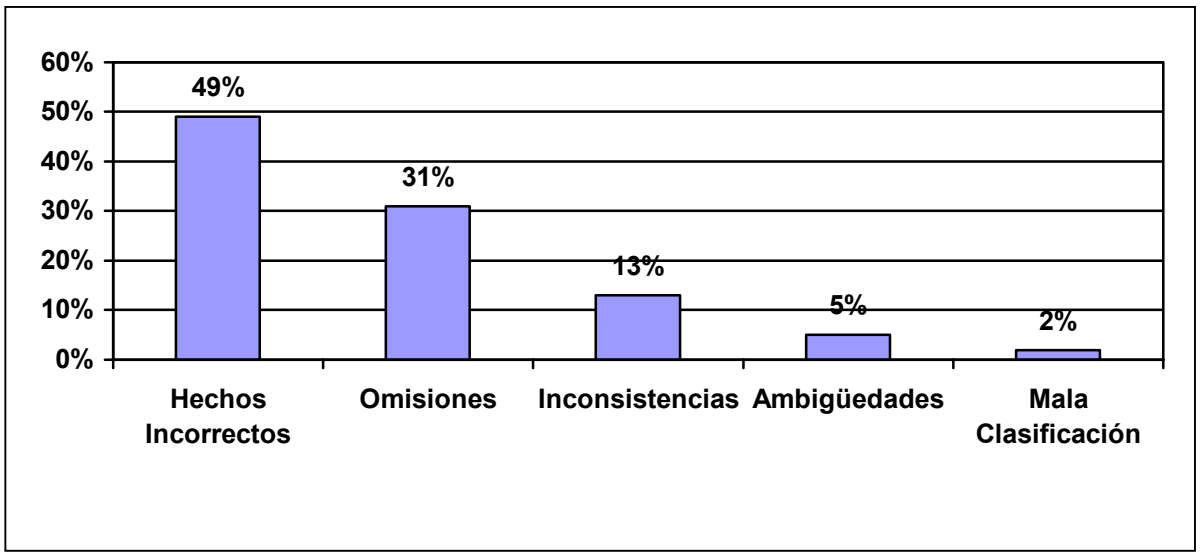


Figura 3-3 Tipos de Errores de Requerimientos [Davis 1993].

Un estudio realizado por la Naval Research Laboratory, demuestra esta falta interés por los productos generados del análisis. El resultado del estudio muestra que el 77% de errores de requerimientos no fueron resultados de labores propias de oficina. Como lo muestra la Figura 3-3, se ha encontrado que estos errores corresponden a hechos incorrectos, omisiones, inconsistencias, ambigüedades y clasificaciones malas de los requerimientos.

Muchas veces estos tipos de errores suceden por creer que no se pueden detectar en la etapa de requerimientos y que su detección es más fácil en las pruebas finales, es decir, con el sistema ejecutándose. Sin embargo, esto no tiene sentido si se toma en cuenta el costo que significa reparar un error en esta etapa.

Además, se ha demostrado que es más efectivo detectar los errores por medio de inspecciones que por cualquier otro medio, es decir, no es necesario esperarse hasta que exista código ejecutable para realizar pruebas (ver Figura 3-4). Esto puede ayudar a detectar los problemas más cerca de su origen reduciendo así los costos. Un beneficio adicional es que las inspecciones permiten establecer rastreabilidad ("trazabilidad") entre los diversos artefactos, modularizar las descripciones, hacer análisis de impacto, etc.

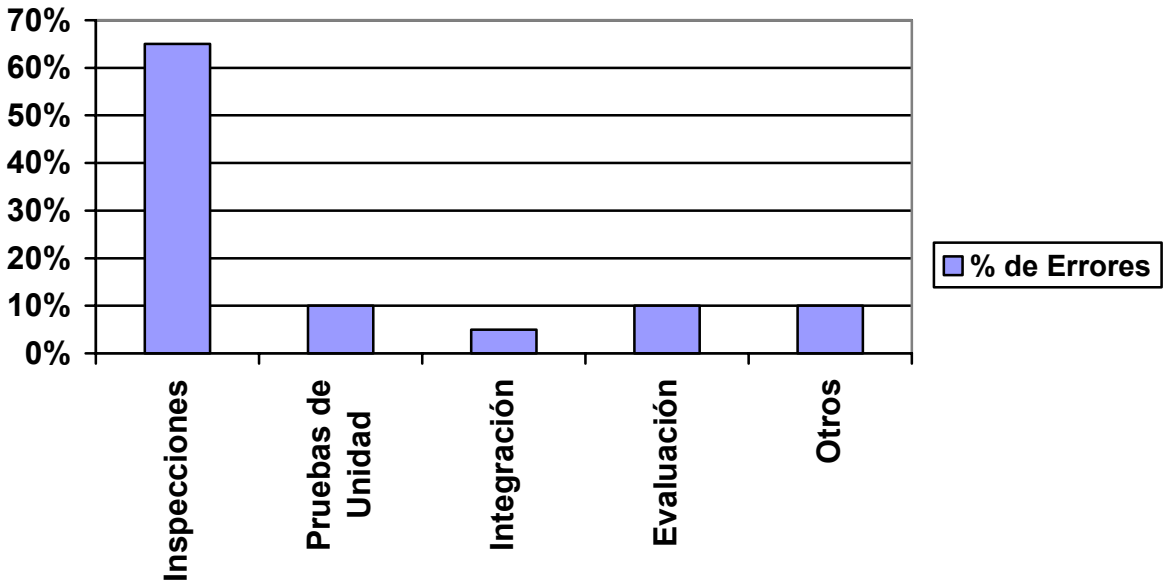


Figura 3-4 Porcentaje de errores detectados según tipo de técnica [Davis 1993]

Desde el punto de vista de un sistema de métricas para el control de los proyectos, la fase de requerimientos debe ser el punto estándar donde empezar el rastreo de costos del proyecto, y el lugar apropiado para iniciar el sistema de calidad y rastreo de errores, ya que los problemas de requerimientos son la mayor fuente de futuros gastos y problemas [Jones 1996b].

El énfasis en los requerimientos no solo debe limitarse a la etapa de análisis, sino que también se debe considerar los requerimientos agregados posteriormente. Si se ve la Figura 3-5, estos representan un incremento en promedio del 15.37% después de concluida la etapa de análisis. Este tipo de requerimientos contribuye a incrementar la diferencia entre los tiempos y presupuestos planeados y reales.

Por esta razón no se debe menospreciar la labor del ingeniero de requerimientos y la actualización y administración del documento de requerimientos, posterior a la finalización de la etapa de análisis.

Puntos de Función	Promedio
1	0.09
10	0.71
100	6.63
1000	18.34
10000	33.66
100000	49.35
Promedio	15.37

Figura 3-5 Incremento porcentual de requerimientos de acuerdo con el tamaño de la aplicación en términos de puntos de función. Se supone que los requerimientos son el 100% después de finalizada la etapa de análisis. [Jones 1996b]

4. El documento de requerimientos

En un proyecto informático, de todas las cosas a documentar, los requerimientos son lo más importante a largo plazo y las especificaciones lo son a corto plazo[7]. La especificación le indica al programador y a los encargados de pruebas exactamente qué hacer y qué probar, sin embargo, con esta información no tienen la pericia para tomar las decisiones de forma inteligente o proponer nuevas ideas para desarrollo futuros. Este problema es superable mediante los requerimientos, los cuales son la única fuente de información para conocer el dominio del problema [Kovitz 1999].

Cuando en un proyecto informático se encuentran involucradas dos o más personas, ocurre la comunicación oral. Durante la transmisión de la información, cada persona la distorsiona un poco y le añade detalles [Kovitz 1999]. El mensaje final puede llegar a ser muy distinto al original.

En cualquier proyecto informático siempre existirá esa comunicación oral, la cual es esencial. Sin embargo, utilizar únicamente este medio para transmitir los requerimientos puede provocar un caos en el proyecto. Un documento de requerimientos adecuado supera esta dificultad, por lo cual no se le puede considerar simplemente un formalismo o un "entregable" de la etapa inicial, pues su propósito es solventar las debilidades que presenta la tradición oral y la memoria humana.

4.1. Beneficios de la documentación en el software

La percepción general hacia el trabajo en la informática está relacionada directamente con las computadoras, por lo cual, a las personas externas a la profesión les es difícil imaginar que una gran parte del trabajo se concentre en "tareas de escritorio", como lo es la documentación.

Cuando un informático, encargado de entregar un producto de software, no está enfrente del computador codificando, la percepción de la mayoría es que no está siendo efectivo o productivo. Este es un problema de cultura informática, el cual es difícil de solucionar, ya que los mismos profesionales del área tienden a pensar de manera similar.

La documentación en el desarrollo de sistemas es un elemento tan importante como el mismo producto codificado. Un proyecto desarrollado sin documentación es susceptible al fracaso o posee calidad desconocida.

Los informáticos conocen este concepto, sin embargo, muy pocos lo aplican. La razón de fondo es el desconocimiento de los beneficios de la documentación en general. Algunos de estos son los siguientes [Kovitz 1999]:

[7] Este caso se refiere a las especificaciones de diseño.

a) **Extiende lo que la mente puede captar y recordar**: En cualquier proyecto, la cantidad de información involucrada es más de lo que una persona puede retener, más aún después de reuniones de ocho horas que cubren cada detalle. Cuando existe documentación, siempre se puede referir a lo escrito y no se pierde, contrario a la memoria humana.

b) **Presenta la misma historia a cada miembro del equipo:** Un documento es igual cada vez que se lee. De esta forma, el diseñador de interfaces, los programadores, los encargados de documentación pueden leer el mismo material, el cual, no sería igual si a todos se les brinda la información por medio de conversaciones individuales.

c) **Introduce nuevos miembros al equipo:** En los proyectos donde existe rotación de personal, la documentación minimiza el riesgo de la "curva de aprendizaje" y los tiempos de adaptación. Por ejemplo, un nuevo miembro de control de calidad, puede invertir mucho tiempo tratando actualizarse por medio de la comunicación oral, sin lograr los resultados adecuados. Un documento de requerimientos bien escrito puede ayudarlo a actualizarse en poco tiempo.

d) **Protege el valor intelectual:** Es común que solo una o dos personas en una compañía conozcan el dominio del problema o el diseño del software. Ellos son los únicos que pueden juzgar de forma inteligente los cambios propuestos, notar faltantes en el razonamiento acerca de la aplicación o pensar en nuevas ideas para el software. Si estas personas escriben su conocimiento, la compañía no será dependiente de éstas. La propiedad intelectual no se perdería si obtienen una mejor propuesta de trabajo.

e) **Ayuda al escritor a entender mejor el problema**: Escribir los requerimientos fuerza a adoptar un estándar de rigor mas allá que la conversación. Cualquiera que ha documentado requerimientos ha tenido la experiencia de descubrir faltantes o incoherencias conceptuales en su entendimiento del problema. Esta es la observación que lleva a la gente a concluir que "el documento no es importante, solo la labor de documentación". Pero como se ha descrito en los otros puntos, también lo es el documento.

4.2. Definición del documento de requerimientos

Los *requerimientos* son los efectos que un computador debe ejercer en el dominio del problema por medio de la virtud de la programación de la computadora [Kovitz 1999]. El *documento de requerimientos* es un enunciado oficial de los *requerimientos* del sistema para los clientes, usuarios finales y desarrolladores del software [Sawyer 1998]. Éste es un acuerdo entre los involucrados acerca de lo que debería ser el futuro sistema.

4.3. Propósitos del documento de requerimientos

El propósito general del documento de requerimientos es proveer un medio para captar y comunicar a todas las partes interesadas qué es necesario. Su objetivo más significativo es servir como un contrato entre el que adquiere y el proveedor de las capacidades, por medio de la definición de qué debe ser provisto y muchas veces la manera en la cual debe ser producido y la tecnología por incorporar [Wilson 1997].

Además, provee una base para la administración del proyecto y las funciones de ingeniería, por ejemplo, evaluar los cambios de propuestos, resolver disputas entre el cliente y el proveedor, desarrollar pruebas a partir de requerimientos, escribir el manual de usuario preliminar, planear actividades de soporte para el mantenimiento, implementar mejoras operacionales [Wilson 1997] y definir versiones del producto.

Estos propósitos no se pueden llevar a cabo a menos que la forma y el contenido del documento lo habiliten para que todos los participantes del proyecto lo entiendan. En la mayoría de los casos estos participantes incluyen el cliente, proveedor, operadores y usuarios[8]. Como estos participantes tienen diferentes perspectivas e intereses de diferentes alcance y profundidad, el documento debe ser muy versátil [Wilson 1997].

4.4. Nivel de especificidad y alcances del documento de requerimientos

Según sea el objetivo del documento de requerimientos, así variará su contenido. Si es para que varios proveedores participen con propuestas, podrá tener menos grado de detalle con el fin de fomentar la competencia. Por el contrario, un documento realizado por los desarrolladores previo al inicio del proyecto tendrá más grado de detalle [Davis 1993].

Esto se relaciona con lo expuesto anteriormente sobre los niveles del documento de requerimientos. En primera instancia se debe definir cuál es el objetivo de éste, para determinar qué debe incluirse. No se puede saber si el contenido corresponde al *qué* del producto, sin definir su propósito.

Por ejemplo, [Rakos 1990] señala en el método de desarrollo dos documentos. Inicialmente, en la etapa de definición se elabora un documento de requerimientos[9], el cual describe el problema del usuario y la solución de manera general. Más adelante, menciona la etapa de análisis, en donde se creará otro documento que describe todo el comportamiento del sistema o en otras palabras, qué hará el sistema para el usuario.

[8] Además de los participantes, también se debe tomar a gente que no está involucrada directamente, pero que sí está interesada en el proyecto.

[9] Este documento de requerimientos tiene el mismo nombre del utilizado en este trabajo. [Rakos 1990] se refiere a éste como una descripción más general del sistema. En el presente documento se utiliza el nombre para generalizar. No deben confundirse.

El nivel de detalle dependerá de varios factores [Sawyer 1998]:

a) De las prácticas normales de la organización.

b) Si la especificación va a ser o no la base para un contrato de desarrollo.

c) El tipo de sistema por desarrollar.

Si se está desarrollando un producto que es especificado y producido por la misma persona, se puede producir una especificación general. Los detalles se agregarán conforme avance el desarrollo. Por el contrario, si se contrata una empresa externa para desarrollar, la especificación debe ser más minuciosa [Sawyer 1998]. Aunque si se necesita promover la competencia, se puede hacer más general [Davis 1993]

En el caso de una especificación como base de un contrato, el lenguaje utilizado es orientado a los negocios y puede carecer de términos técnicos [Rakos 1990]. Este tipo de documentos tiende a incluir más requerimientos funcionales. En cambio otros más específicos incluyen tanto requerimientos funcionales como no funcionales[10]. Los requerimientos funcionales definen qué debe hacer el sistema, describen todas las entradas y salidas del sistema y sus relaciones. Los requerimientos no funcionales definen los atributos del sistema[11] cuando éste lleva cabo el trabajo. Estos pueden ser niveles de eficiencia, confiabilidad, seguridad, portabilidad, etc. [Davis 1993]

Los documentos escritos con el fin de iniciar un desarrollo, deben ser más específicos. El propósito de este tipo de documento es proveer un medio para [Davis 1993]:

a) Comunicación entre clientes, usuarios, analistas y diseñadores: Un documento de requerimientos bien escrito reduce la probabilidad de que un cliente quede insatisfecho con el producto final, ya que no deberían existir malas interpretaciones. El problema es cuando se deja intencionalmente ambiguo para no perder flexibilidad en el diseño, lo cual incrementa el riesgo del proyecto. Un documento de requerimientos debe ser muy específico en cuanto al comportamiento externo del sistema. Si es necesario incluir un diseño para explicar el comportamiento, se debe indicar que éste es únicamente de carácter explicativo y no debe interpretarse como el diseño.

b) Soporte de las actividades de pruebas de sistema: El segundo propósito de un documento de requerimientos es servir de base para la prueba de sistemas y las actividades de validación.

[10] [Davis 1993] señala estos requerimientos como de comportamiento y no de comportamiento. En vez de estos se utilizaran los términos funcionales y no funcionales, como los define [Sawyer 1998].
[11] Algunos autores llaman “atributos de calidad” a los requerimientos no funcionales.

c) Controlar la evolución del sistema: Cada vez que se agregue, elimine o modifique un requerimiento, se debe cambiar el documento de requerimientos. Estas modificaciones deben ser controladas por el proceso de administración de la configuración para poder tener un registro formal de la evolución del proyecto.

Cualquiera que sea el estilo o grado de especificidad del documento de requerimientos, este no debe contener lo siguiente [Davis 1993]:

- Requerimientos del proyecto (recurso humano, tiempos, costos, etc.): No se debe incluir esta información porque el documento de requerimientos tiene una longevidad (perdurabilidad) distinta a los datos del proyecto. El documento de requerimiento es parte del producto final y por lo tanto tiene la misma vida que la utilización de éste. Por el contrario, la información del proyecto solo tiene importancia durante el desarrollo del proyecto[12].

- Diseños: Existen varias razones para mantener el diseño separado del documento de requerimientos:

  a) El tener el diseño separado del documento de requerimientos tiene la ventaja de poder medir mejor el progreso y mostrar resultados de una forma más temprana. Además, se puede utilizar para controlar los cambios en el sistema.

  b) Existe una diferencia entre las audiencias del documento de requerimientos y el diseño. En el primer caso incluye usuarios del sistema, encargados de pruebas del sistema, clientes, diseñadores y los ingenieros de los requerimientos. Por otro lado, el diseño incluye a los encargados de prueba de unidad e integración, programadores y diseñadores.

  c) Como consecuencia de la división de funciones, los ingenieros de requerimientos tienen habilidad en análisis y especificación, y no necesariamente en aspectos de diseño.

- Planes de aseguramiento del producto: Existen dos razones para no incluir planes de aseguramiento de calidad dentro del documento de requerimientos:

  a) Son temas distintos, a pesar de que el documento de requerimientos es fuente para los planes de validación y verificación. Además, al igual que el diseño, el tiempo de vida de este es distinto a los requerimientos.
  b) Las audiencias son distintas.

[12] En un sentido estricto, esta información sirve para fundamentar futuras estimaciones, formar bases de conocimientos sobre riesgos, elaborar plantillas de proyectos, formular planes de contingencia, etc.

5. El proceso de definición de requerimientos

Se ha mencionado mucho los requerimientos y el documento asociado, pero aún no se ha dicho quién, ni cuándo se debe producir el documento. Es obvio que, si los requerimientos son necesarios para definir qué es lo que debe hacer el sistema, si no se les ha definido, no se puede iniciar la producción de sistemas.

Todo método de desarrollo, razonablemente completo, contiene una etapa inicial que corresponde a la definición de los requerimientos. En ésta ocurren dos actividades importantes: El *análisis del problema* y la *descripción del producto*, las cuales no necesariamente son secuenciales y ni mutuamente excluyentes [Davis 1993].

Durante el *análisis del problema*, se invierte tiempo entrevistando personas que tienen el mayor conocimiento acerca del problema, haciendo tormenta de ideas e identificando todas las posibles restricciones en la solución del problema. En esta actividad existe un considerable aumento de la información y conocimiento acerca del problema. Esta etapa finaliza cuando se tiene un completo entendimiento del problema [Davis 1993].

Durante la *descripción del producto* se elabora el documento que describe el comportamiento externo esperado del producto a construir, para resolver el problema que fue entendido durante el análisis. En esta actividad se organizan las ideas, se resuelven puntos de vista conflictivos y se eliminan inconsistencias y ambigüedades [Davis 1993].

5.1. Análisis del problema

A pesar que este trabajo está fuertemente relacionado con la etapa de análisis, no es su intención investigar sobre los métodos utilizados para el análisis del problema. Sin embargo, es importante profundizar sobre el espacio del producto y las primitivas de esta actividad.

La actividad del análisis del problema incluye los siguientes aspectos [Davis 1993]:

a. Aprender acerca del problema.

b. Entender las necesidades de los usuarios potenciales.

c. Tratar de encontrar quién realmente es el usuario.

d. Entender todas las restricciones en la solución.

Suponiendo que el proceso de definición requerimientos termina cuando se crea un *documento de requerimientos*, el cual contiene una completa descripción del comportamiento externo del producto a construir o comprar, la actividad de *análisis del problema* se puede describir como la encargada de definir el *espacio del producto*, esto es, el conjunto de posibles soluciones de software que satisfacen todas las restricciones conocidas [Davis 1993].

Las restricciones a las soluciones de un problema están determinadas por [Davis 1993]:

a) Usuarios: Los usuarios pueden conocer muy bien el problema y lo que quieren; sin embargo, esto no necesariamente puede corresponder a lo que ellos realmente necesitan. Por ejemplo, los usuarios pueden querer un sistema que no modifique sus actividades no automatizadas, pero lo que ellos realmente necesitan es un cambio radical en la manera de realizar negocios. El trabajo del analista[13] consiste en distinguir estas diferencias, con las cuales puede dividir el universo de sistemas en dos conjuntos mutuamente excluyentes, aquellos que cumplen con las necesidades del usuario y aquellos que no[14].

b) Clientes: Estos corresponden a las personas que tienen los recursos para hacer la compra o subvencionar el producto. Pueden ser los mismos usuarios. Los requerimientos que le interesan a este grupo pueden incluir: funciones llevadas a cabo, tiempo de desarrollo, costos, mantenibilidad, modificabilidad y confiabilidad. El analista debe dividir el universo de posibles soluciones en aquellas que cumplen con estos requerimientos y los que no.

c) Desarrolladores[15]: La organización desarrolladora, además de querer satisfacer los requerimientos de los usuarios y los clientes, se interesa en tener ganancias, mercado, una relación del producto con productos existentes y la habilidad de su gente para realizar el trabajo propuesto. Dependiendo de la relación entre el cliente y el desarrollador, pueden surgir otros factores. De acuerdo con los requerimientos de la empresa desarrolladora, el universo se puede dividir en aquellos desarrollos factibles y los que no.

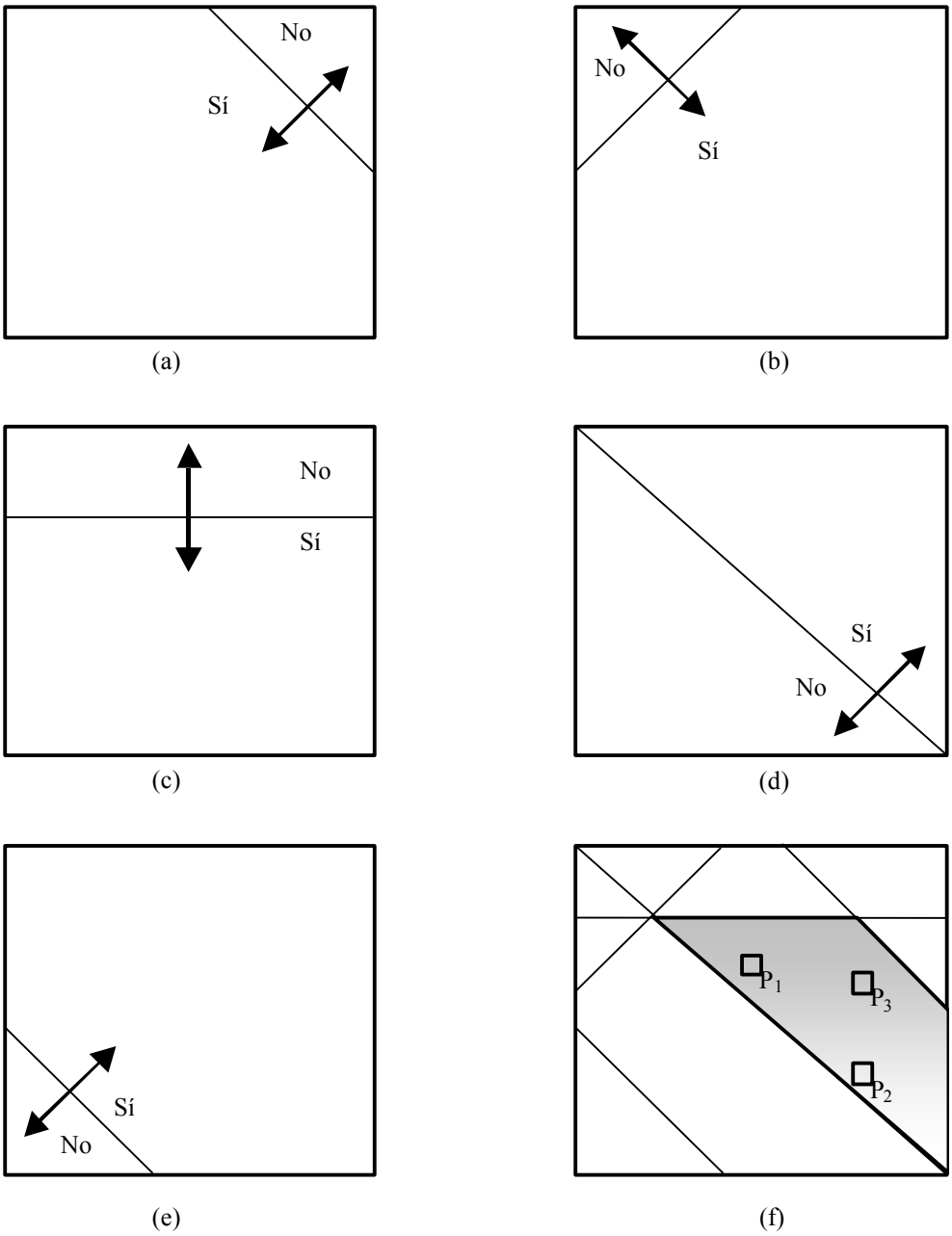
d) Tecnología: La tecnología también tiene un impacto sobre el espacio del producto. Se debe distinguir en las soluciones el riesgo asociado al uso de tecnología (sea muy nueva o muy antigua) para dividir el espacio del producto en riesgos aceptables y los inaceptables.

e) Estándares: Las leyes y los estándares también pueden restringir el espacio del producto. Puede ser que una solución viole algún estándar o ley, por lo cual no es factible su implementación.

[13] Analista se refiere a la persona encargada del proceso de definición de requerimientos.
[14] Hay quienes proponen que los analistas "inventen requerimientos", para lograr sistemas más competitivos e innovadores [Robertson 2002].
[15] La organización desarrolladora puede ser externa o interna. En este último caso, un departamento ofreciendo los servicios a otros dentro de la misma organización.

La intersección de todos estos conjuntos de soluciones factibles formará el *espacio del producto,* el cual define el dominio de los productos que satisfacen todas las restricciones de la solución. Dentro de este dominio, cualquier producto será un producto aceptable (Figura 5-1). Vale decir que entre los productos aceptables podría haber algunos más "deseables" que otros (por precio, flexibilidad, simplicidad, etc.).

(Por su tamaño, la figura aparece en la página siguiente.)



Construyendo el espacio del producto: (a) Necesidades de usuario; (b) Necesidades del Cliente; (c) Perspectiva del Desarrollador; (d) Riesgos Tecnológicos; (e) Impacto de las leyes y los estándares; (f) La parte sombreada representa el espacio del producto. Cualquier producto P_i que se encuentre en este espacio, es considerado un producto aceptable.

Figura 5-1 Espacio del Producto

Es importante mencionar que, para la mayoría de los problemas, al delinear inicialmente todas las restricciones puede resultar en un espacio del producto de área negativa, es decir, dos o más restricciones se contradicen. La actividad más difícil en el análisis es precisamente lograr que todas estas restricciones se concilien. Ejemplos comunes de esto son las negociaciones entre funcionalidad y costo, o entre funcionalidad y tiempo de desarrollo [Davis 1993]

Durante la actividad de *análisis del problema*, todos los aspectos del dominio del problema deben ser investigados. Estos, independiente del tipo de problema presentado, se relacionan siempre con *objetos*, *funciones* y *estados*, los cuales pueden ser descritos con múltiples niveles de abstracción o detalle, y con diverso número de relaciones [Davis 1993].

Para organizar toda esta información, los analistas tienden a usar técnicas que les permiten organizar el conocimiento, en particular el *conocimiento estructurado*, el cual es un conjunto estructurado de conceptos y sus relaciones [Davis 1993].

En la estructuración del conocimiento, se utilizan tres principios para llevar a cabo el análisis; estos son [Davis 1993]:

a. *Particionamiento*: Involucra la relación "es parte de / agregación" entre objetos, funciones o estados. Por ejemplo, "el motor es parte de un carro".

b. *Abstracción*: Involucra la relación "general / específico", "ejemplo de" o "instancia de" entre objetos, funciones o estados. Por ejemplo, "el vehículo es terrestre, aéreo o marítimo".

c. *Proyección*: Especifica la relación entre "puntos de vista" entre objetos, funciones o estados. Por ejemplo, "el punto de vista del chofer o del pasajero".

Estos conceptos tienden a mostrarse de forma jerárquica, como lo muestra la Figura 5-2. La mayoría de las técnicas conocidas para el análisis del problema utilizan alguna de estas formas primitivas para representar las relaciones entre funciones, estados y objetos.

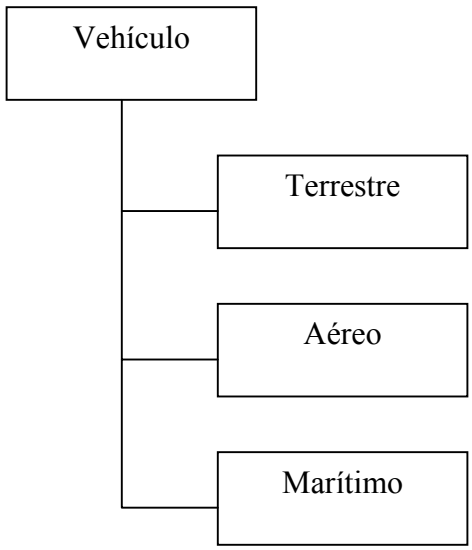


Figura 5-2 Ejemplo de abstracción

Aunque la actividad que más compete a este trabajo está relacionada con la descripción del problema y más específicamente al producto final generado, el *documento de requerimientos,* es importante conocer cómo se debe definir el *espacio del producto* y cuáles son las formas básicas para organizar el conocimiento. Esto es la entrada fundamental para la definición de un *documento de requerimientos*.

5.2. Descripción del producto

El proceso de definición de requerimientos no se puede dar por concluido hasta que no se tenga un *documento de requerimientos*, el cual es una declaración oficial de los requerimientos para los clientes, usuarios finales y los desarrolladores del software [Sawyer 1998]. El documento de requerimientos es esencial para la planificación, el desarrollo y calidad del producto final.

La actividad de *descripción del producto* es donde el analista realiza la representación documentada de las condiciones y capacidades que debe considerar el producto final. La mayoría de estos requerimientos son escritos como oraciones en lenguaje natural complementadas con diagramas y tablas con información detallada [Sawyer 1998].

Al menos tres problemas surgen del uso del lenguaje natural para describir sistemas complejos y dinámicos: ambigüedad, inexactitud e inconsistencia. Si se trata de definir grandes capacidades multi-dimensionales dentro las limitaciones impuestas por la estructura bi-dimensional de un documento, se pueden ocultar o desfigurar las relaciones entre los grupos de requerimientos [Wilson 1997].

Estos problemas son inherentes a cualquier lenguaje natural. Cuando se habla, los tonos, el movimiento de las manos y el lenguaje corporal, puede ayudar a clarificar un concepto [Davis 1993], sin embargo, la forma escrita del lenguaje puede ser más ambigua, por lo que se debe tener más cuidado.

A pesar de las ventajas atribuidas al uso de lenguajes de especificación formal sobre el lenguaje natural, su uso no ha sido una práctica común. Las especificaciones comúnmente son escritas en lenguaje natural porque los requerimientos tienen que ser entendidos tanto por el cliente como por el desarrollador [Wilson 1997].

No existe una manera óptima de escribir requerimientos, pues depende de las prácticas normales de la organización y de las notaciones que son usadas por los escritores y redactores de los requerimientos [Sawyer 1998]. Sin embargo, precisamente uno de los énfasis del presente trabajo es ofrecer lineamientos que permitan al analista crear un documento con características que reduzcan el riesgo que existan estas ambigüedades, inexactitudes e inconsistencias.

Es importante tratar de generar de esta actividad un documento de calidad[16], ya que es el primer producto tangible de la mayoría de ciclos de vida y la mayor fuente de problemas en las fases posteriores [Wilson 1997].

[16] Los atributos para determinar la calidad de los requerimientos serán vistos en una sección posterior.

6. La ingeniería de requerimientos

La *ingeniería de requerimientos* es la actividad que se encarga de descubrir, documentar y mantener un conjunto de requerimientos para un sistema basado en computadoras [Sawyer 1998]. La utilización de este término en vez de *análisis de sistemas* tiene dos razones principales:

a) El análisis de sistemas se asocia directamente con la etapa inicial de análisis, excluyendo la labor de administración de los requerimientos, que implica el control sobre la modificación, eliminación, inclusión y negociación de los requerimientos posterior a la etapa de inicial de definición, asegurándose siempre que el documento de requerimientos resultante de este proceso cumpla con los atributos de calidad.

b) El término ingeniería implica el uso de técnicas sistemáticas y repetitivas que aseguren el cumplimiento de los requerimientos con los atributos de calidad [Sawyer 1998].

No todas las organizaciones tienen el mismo nivel de ingeniería en el proceso de requerimientos; algunas aplican prácticas más avanzadas. Es imposible sugerir lineamientos a una empresa para mejorar este proceso, sin saber la situación en que se encuentra.

Para esto, [Sawyer 1998] propone un modelo basado en el CMM[17], para definir la madurez actual de las organizaciones con respecto de los requerimientos. Este proceso de madurez es el nivel en que una organización tiene definido el proceso de ingeniería de requerimientos basándose en las buenas prácticas de la ingeniería de requerimientos.

Los niveles son los correspondientes:

Nivel 1: Nivel inicial

- En este nivel las organizaciones no tienen definido un proceso de ingeniería de requerimientos.
- Estas no usan métodos avanzados para soportar este proceso.
- Muy comúnmente no se producen documentos de requerimientos de buena calidad en el tiempo y presupuesto establecidos.
- Dependen en las habilidades y experiencia de los ingenieros para el descubrimiento, análisis y validación de los requerimientos.

[17] Capability Maturity Model (CMM) del Software Engineering Institute (SEI).

Nivel 2: Nivel repetible

- En este nivel las organizaciones tienen definidos y descritos estándares para los documentos de requerimientos.
- Introducen políticas y procedimientos para la administración de los requerimientos.
- Pueden usar herramientas avanzadas y técnicas en su proceso de ingeniería de requerimientos.
- Los documentos de requerimientos son, por lo general, de alta calidad y son producidos dentro de los tiempos establecidos.

Nivel 3: Nivel definido

- En este nivel las organizaciones tienen un modelo del proceso de ingeniería de requerimientos basado en buenas prácticas y técnicas.
- Tienen un programa de mejoramiento continuo y pueden hacer la valoración de objetivos de acuerdo con nuevos métodos y técnicas.

Intuitivamente, se puede afirmar que en nuestro medio la mayoría de las empresas se encuentran en el primer nivel, por lo que el presente trabajo puede ser de gran utilidad para mejorar la madurez de las organizaciones en el área de ingeniería de requerimientos.

7. Calidad en los documentos de requerimientos

Aunque las especificaciones formales proveen gran rigurosidad y precisión, son pocos los desarrolladores de software, y virtualmente ningún cliente, quienes estén familiarizados con notaciones formales. A pesar de sus desventajas, el lenguaje natural estructurado se mantiene como la vía más práctica de documentar requerimientos para la mayoría de los proyectos [Wiegers 1999].

Las propiedades del lenguaje natural merecen una atención particular, ya que es por este medio que un documento de requerimientos cumple con su propósito principal, el cual es llevar la información, obtenida del cliente y de otras fuentes, a los desarrolladores [Fabbrini 1998].

Para ayudar a la comprensión, análisis y control del concepto de calidad en los documentos de requerimientos, es necesario definir un modelo por medio del cual se establezcan las características a cumplir. Tradicionalmente, estos modelos corresponden a listas de propiedades y no establecen una clasificación de los atributos lingüísticos relevantes.

En esta sección se presenta el modelo planteado por [Fabbrini 1998], el cual es basado en el modelo presentado por Krogstie, Linland y Sindre en la sétima Conferencia Interacional CAiSE. El modelo de Krogstie[18] es un enfoque para el análisis sistemático de aspectos de calidad; el de [Frabbrini 1998] es una propuesta adaptada para el análisis de las propiedades de calidad de los documentos de requerimientos.

[18] Por facilidad se menciona a únicamente a Krogstie, pero el modelo fue elaborado por Krogstie, Lindland y Zinder.

7.1. El modelo de calidad de Krogstie, Lindland y Zinder [Fabbrini 1998]

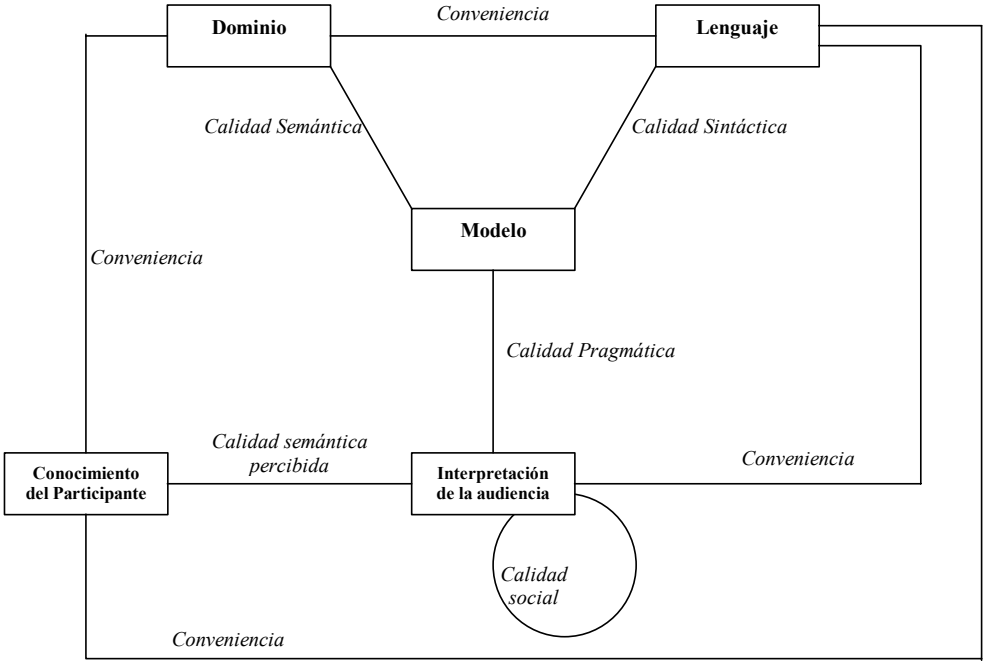


Figura 7-1 Modelo de calidad de Krogstie, Lindland y Sindre

Los principales conceptos del este modelo de calidad son (Figura 7-1):

A, la audiencia[19]: Es el conjunto de individuos y actores organizacionales, llamados también participantes, y los actores técnicos (sistemas y herramientas), los cuales necesitan entender el modelo.

M, el modelo: Es el conjunto de todos los enunciados explícitos e implícitos realizados por la audiencia acerca del problema a mano. Para cada participante en la audiencia, la parte del modelo considerado relevante para éste, puede ser vista como la proyección del modelo total.

L, el lenguaje: Es el conjunto de todos los enunciados que pueden ser hechos de acuerdo con el vocabulario y la gramática del modelo de lenguaje usado. Varios lenguajes (formales o informales) pueden ser usados al mismo tiempo.

[19] La audiencia no está representada en la Figura 7-1, lo que se diagrama es la interpretación de ésta hacia el modelo (*I*).

D, el dominio: Es el conjunto de todos los enunciados que podrían ser correctos y relevantes al problema en cuestión. D denota el conocimiento ideal acerca del problema y se desarrolla durante el proceso de ingeniería de requerimientos. Se debe notar que D concierne con el problema, no con la solución.

I, la interpretación de la audiencia: Es el conjunto de enunciados que, según la audiencia, conforman el modelo.

K, Es el conocimiento de los participantes: Es el conjunto de todos los enunciados que serían ser correctos y relevantes para resolver el problema en cuestión de acuerdo con el conocimiento de los participantes.

Empezando por los componentes básicos involucrados en el proceso de ingeniería de requerimientos, Krogstie ha propuesto una serie de tipos de calidad, los cuales se originan de las relaciones entre los distintos componentes descritos anteriormente. Los tipos de calidad son *sintáctica, semántica, semántica percibida, pragmática y calidad social*. Además, son establecidas algunas relaciones de conveniencia. Cada tipo de calidad ha sido refinado en un conjunto de propiedades llamadas *metas*. Para cada meta, se ha propuesto una colección de *medios* para lograrlos, consistiendo de *propiedades del modelo* y *actividades* que influyen en el modelo de actividades. Para mantener el estándar de la terminología de calidad, se usará el nombre *factor* en vez de *meta*, y *criterio* en vez de la *propiedad del modelo*.

- La *Calidad Sintáctica* es la correspondencia entre el modelo y el lenguaje utilizado para expresarlo.

- La *Calidad Semántica* es la correspondencia entre el modelo y el dominio. Si $M \setminus D \neq \emptyset$, entonces el modelo contiene enunciados inválidos. Si $D \setminus M \neq \emptyset$, entonces el modelo es incompleto. Como no es probable alcanzar la total validez y completitud, Krogstie, introducen las nociones de *validez factible* y *completitud factible,* como canjes entre los beneficios y costos de quitar y agregar enunciados al modelo.

- La *Calidad Pragmática* es la correspondencia entre el modelo y la interpretación de la audiencia. Su propósito principal es la comprensión, esto es, que el modelo sea entendido por la audiencia y no el concepto abstracto de comprensión. Como la total comprensión es algo irrealista, la *comprensión individual* es suficiente, esto es, que la interpretación de cada participante coincida con la proyección del modelo que es relevante para él.

- La *Calidad Semántica Percibida* es la correspondencia entre la interpretación del modelo por el participante y su propio conocimiento. Análogamente a la calidad semántica, la validez percibida total y la completitud percibida total son difíciles de alcanzar, por lo cual son introducidas las nociones de *validez percibida factible* y *completitud percibida factible*.

- La *Calidad social* es la correspondencia entra las interpretaciones de los participantes. El objetivo es el acuerdo, o, en su forma débil, el *acuerdo factible*.

Finalmente, las consideraciones de *conveniencia* pueden afectar toda la calidad de un modelo conceptual, incluyendo la conveniencia del lenguaje escogido para describir el problema o el usado por los participantes.

7.2. Modelo adaptado para la lingüística [Fabbrini 1998]

En la adaptación a la lingüística, se consideran solo los elementos del modelo original que están directamente conectados al modelo y al lenguaje (Figura 7-2).

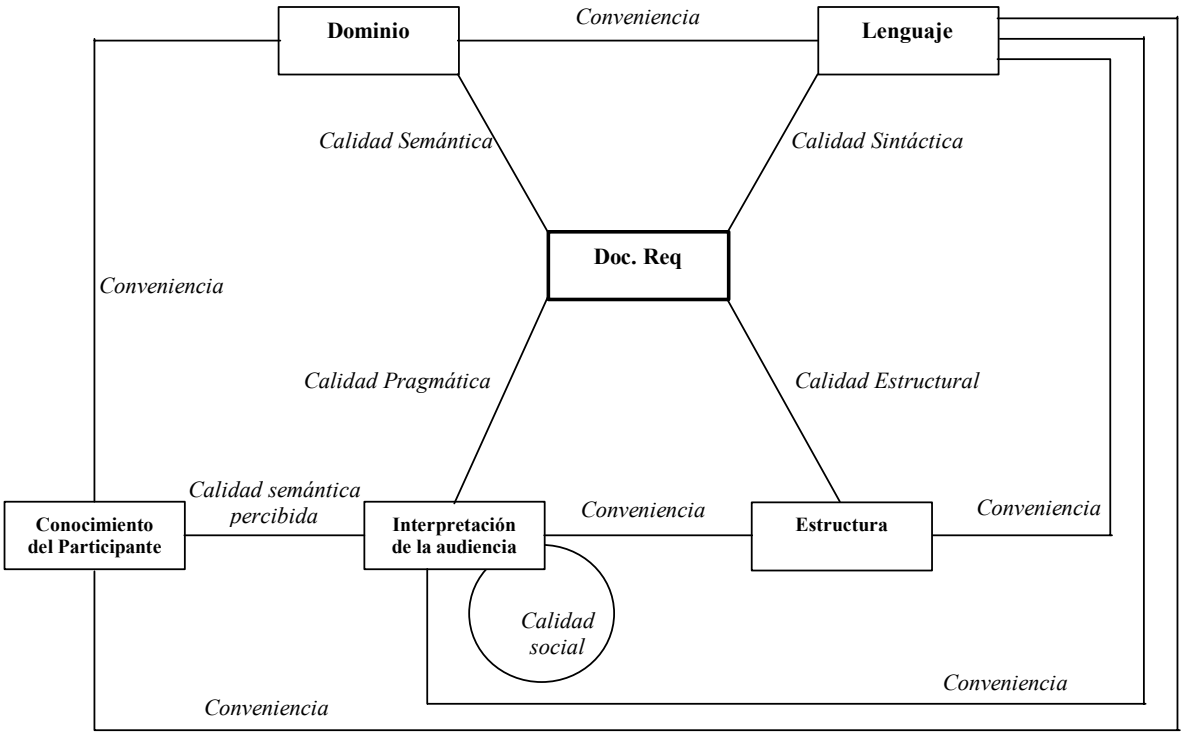


Figura 7-2 Modelo de calidad para requerimientos en lenguaje natural

El lenguaje *L*, es un conjunto de enunciados que pueden ser especificados por una sintaxis formal o por medios informales, por ejemplo, la gramática española. Aún más, *L* tiene una semántica, la cual puede ser forma o informal, por ejemplo, el sentido común. En ambos casos se supone que existe una vinculación que relaciona los conjuntos de enunciados de *L*, con los enunciados implicados por estos.

También se introduce el conjunto de estructura *S*, el cual especifica, formal o informalmente, la estructura de los requerimientos y la de los documentos de requerimientos.

Un *requerimiento* es una colección estructurada de enunciados. Un *documento de requerimientos* *R* (correspondiente al modelo *M* en la Figura 7-1) es una colección estructurada de requerimientos. Cada estructura en *R*, (capítulo, sección, párrafo, lista de enunciados, etc.) es llamada una *unidad estructural* de *R*. *R* normalmente evoluciona durante el proceso de requerimientos, por lo tanto, algunas de sus partes pueden estar vacías en algún punto en el tiempo.

El modelo de [Fabbrini 1998] también difiere del de Krogstie en los siguientes aspectos:

a) Krogstie considera *R* como un conjunto de enunciados, sin ninguna relación entre ellos.

b) Krogstie considera la estructura como una subpropiedad de modificabilidad. [Fabbrini 1998] coloca la estructura en un rol más importante, dada su relevancia en propiedades como la comprensión.

c) [Fabbrini 1998] incluye en el modelo en *R* solo los requerimientos realizados de forma explícita, sin incluir los implícitos, como lo plantea Krogstie. Esto esencial a nivel teórico, debido al supuesto de que existe un vínculo entre los enunciados. Desde un punto de vista práctico, esta definición es más cercana a un documento de requerimientos real y enfatiza en el hecho que las técnicas de procesamiento del lenguaje natural solo tratan enunciados explícitos.

La *audiencia* es definida como "todo aquel que necesita entender el modelo, incluyendo a los patrocinadores del proceso de desarrollo". Dentro de la audiencia se distinguen actores individuales y organizacionales, llamados también participantes, y los actores técnicos. Las definiciones del dominio *D* y de la interpretación de la audiencia *I*, son iguales en ambos modelos.

Una coincidencia en los modelos es que las relaciones con los componentes sintácticos, estructurales, semánticos y pragmáticos con los documentos en lenguaje natural se mantienen.

La calidad social y la calidad semántica percibida, las cuales se concentran en acuerdos sociales y aspectos cognitivos, más que en aspectos de lingüística, no están directamente relacionadas con el documento de requerimientos. Sin embargo, se observa en un sentido más amplio, la calidad social y la semántica percibida como un componente lingüístico, que es afectado por la calidad sintáctica, estructural, semántica y pragmática.

Las relaciones de conveniencia tampoco están directamente relacionadas con el documento de requerimientos. De hecho, en un proceso real, la selección del lenguaje, estructura, audiencia y dominio, solo son relevantes en la fase precedente al análisis de requerimientos. Es interesante notar que en cualquier proceso, la selección de cualquiera de estos cuatro elementos puede estar sujeta a restricciones organizacionales, mientras que el resto pueden ser seleccionados de forma acorde.

7.3. Tipos de calidad [Fabbrini 1998]

De acuerdo con el modelo, se distinguen cuatro tipos de calidad lingüística: *sintáctica, estructural, semántica* y *pragmática*, las cuales corresponden a las relaciones establecidas anteriormente.

a) La *calidad sintáctica* es la correspondencia entre los enunciados en R y el lenguaje L.

b) La *calidad estructural* es la correspondencia entre las unidades en R y la estructura en S.

c) La *calidad semántica* es la correspondencia entre los requerimientos en R y el dominio D.

d) La *calidad pragmática* es la correspondencia entre R y su interpretación por la audiencia, I.

Para lograr estos tipos de calidad, existe una relación de precedencia. Por ejemplo, la calidad sintáctica es un prerrequisito para la calidad semántica y pragmática. La Figura 7-3 muestra los niveles ascendentes para alcanzar los objetivos de calidad. Como consecuencia, las actividades que afectan a un factor, también afectan los factores de los tipos de calidad de más alto nivel. Desde un punto de vista práctico, las relaciones de precedencia sugieren que un programa de mejoramiento de la calidad en una organización debe estar estructurado para lograr los tipos de calidad en el orden mostrado, empezando desde la calidad sintáctica hasta la social.

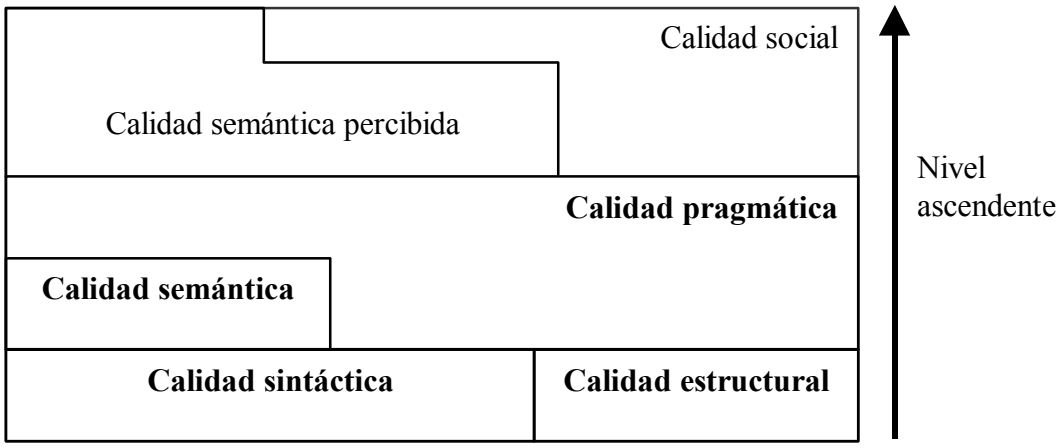


Figura 7-3 Precedencias entre los tipos de calidad

7.4. Factores [Fabbrini 1998]

Los tipos de calidad se pueden refinar en factores considerando, donde tenga sentido, las relaciones de correspondencia que se mencionaron anteriormente: *validez* y *completitud*.

La *calidad sintáctica* se refina en el factor de *validez sintáctica*, esto es, la propiedad que los enunciados en R se encuentren en L. El factor de completitud sintáctica, es decir, que la propiedad que R contenga todos los enunciados en L, no tiene sentido y, por lo tanto, no está incluido en el modelo.

La *calidad estructural* se refina en los factores de *validez estructural* y la *completitud estructural*. La primera consiste en la propiedad que cada unidad estructural en R sea una estructura legal según S. La segunda es la propiedad que las unidades estructurales requeridas en S se encuentren realmente presentes en R.

La *calidad semántica* se refina en *validez semántica* y *completitud semántica*. La primera es la propiedad de que los enunciados en R estén en el dominio D, en otras palabras, que sean correctos y relevantes al problema en cuestión. La completitud semántica es la propiedad que los enunciados en D se encuentren en R, o que puedan ser deducidas por R mediante la vinculación de L. En otras palabras, que el documento de requerimientos contenga o implique cada enunciado correcto y relevante acerca del problema en cuestión.

Los factores de la *calidad pragmática* son la *comprensión válida*: la propiedad de que los enunciados, en la interpretación I de la audiencia se encuentren realmente en R, y la *comprensión completa*: la propiedad que los enunciados en R se encuentren en I. En otras palabras, la comprensión válida significa que todo lo que ha sido entendido por los participantes

es un enunciado en *R*, mientras que comprensión completa, significa que cada enunciado en *R* ha sido entendido por al menos un participante. En general, la total coincidencia entre *R* e *I* no es realista; aún en los problemas más simples, no se puede suponer que cada uno de los participantes entiende cada parte de *R*. La medida en que *R* e *I* corresponden tiene que ser determinada en la definición de los objetivos de calidad para un documento de requerimientos. Como en el modelo original de Krogstie, la comprensión individual es usualmente suficiente.

A diferencia de Krogstie, [Fabbrini 1998] no define la noción de factibilidad debido a que el análisis de factibilidad ocurre naturalmente en un modelo de calidad cuando se determinan los objetivos de calidad esperados.

7.5. Utilidad del modelo

Un proceso de calidad no se puede lograr si no se establecen claramente cuáles serán los criterios para determinar el nivel de calidad del producto. En el caso de los documentos de requerimientos, es fácil incurrir en el error de creer que con un conjunto de técnicas o herramientas, es suficiente para lograr la calidad del documento debido, en gran parte, al desconocimiento del tipo de calidad que se busca en el documento.

Las diferentes técnicas que existen para apoyar la creación de documentos de calidad están orientadas a satisfacer uno o varios criterios. Los criterios son propiedades del documento de requerimientos, que por un lado refinan los factores y por otro abstraen la funcionalidad común de diferentes técnicas. Mientras los factores expresan las necesidades, los criterios son las propiedades ingenieriles del documento de requerimientos.

El modelo de [Fabbrini 1998] es una herramienta útil para poder determinar si los criterios que se desean evaluar cubren todos los aspectos de calidad. Además, la clasificación por tipos de calidad, puede ayudar a definir una estrategia para implementar un modelo de calidad en los documentos de requerimientos.

La Figura 7-4 muestra un ejemplo de actividades para cada uno de los criterios de los factores [Fabbrini 1998]. Cada criterio tiene asociadas dos actividades, una orientada a evaluar la calidad y otra para ser inducida durante el proceso de producción de requerimientos. En este caso, una actividad es la aplicación de una técnica como un medio para controlar o verificar alguna propiedad del documento de requerimientos.

Tipos de Calidad	Factores	Criterio	Actividades
Sintáctica	Validez Sintáctica	Correctitud del léxico	Verificación del léxico Vocabulario controlado
		Correctitud Sintáctica	Verificación de la sintaxis Escritura orientada a la sintaxis
Estructural	Validez Estructural	Correctitud de la estructura	Comprobación de la estructura Plantillas
	Completitud estructural	Presencia de la unidad estructural	Listas de cotejo Plantillas
		Modificabilidad de la estructura	Comprobación de referencias Rastreo
Semántica	Validez Semántica	Correctitud independiente del dominio	Comprobación de ambigüedad Escritura orientada a la semántica
		Correctitud dependiente del dominio	Modelado basado en el dominio Escritura orientada a la semántica
	Completitud Semántica	Completitud independiente del dominio	Comprobación de consistencia Modelado "Speech Act"
		Completitud dependiente del dominio	Razonamiento por casos Diálogos inteligentes
Pragmático	Comprensión válida	Orden de la presentación	Comprobación de referencias hacia delante Modelado estructural
		Legibilidad	Fórmulas de legibilidad Guías de estilo
	Comprensión completa	Navegación	Navegación ("browsing") Edición de hipertexto
		Conciso	Tamaño del documento Limitación del tamaño

Figura 7-4 Ejemplo de criterios y actividades relacionados a los tipos y factores de calidad del documento de requerimientos [Fabbrini 1998]

8. Atributos de calidad de los requerimientos

Con el fin de entender los problemas que padecen los documentos de requerimientos y para encontrar una manera de prevenirlos, es necesario especificar cuáles son los requerimientos que los propios requerimientos deben satisfacer. A estos se les llama *atributos de calidad* de los requerimientos, o como los identifica [Fabbrini 1998], *criterios*.

8.1. Atributos generalmente aceptados

La mayoría de los autores coincide sobre cuáles deben ser estos atributos, sin embargo, existen algunas diferencias. En esta sección se describen los atributos o características que varios autores ([Davis 1993], [IEEE 1998] y [Wilson 1997]) consideran deben satisfacer los requerimientos y que corresponden a los generalmente aceptados para evaluar el documento de requerimientos.

Se considera que un documento de requerimientos debe ser:

8.1.1. Correcto

([Davis 1993], [IEEE 1998], [Wilson 1997])

Un documento de requerimientos es *correcto* si, y solo si, cada requerimiento declarado representa algo requerido del sistema por construir [Davis 1993]. Para esto, el documento debe identificar de forma exacta y precisa las condiciones y limitaciones individuales de todas las situaciones con las cuales se encontrarán las capacidades (del sistema) deseadas y debe definir la respuesta adecuada de la capacidad a esas situaciones. En otras palabras, la especificación debe definir el ambiente operacional del mundo real de la capacidad, su interfaz a ese ambiente y su interacción con ese ambiente [Wilson 1997].

En estos momentos no existe herramienta capaz de asegurar este atributo [IEEE 1998]. Si un programa debe responder a todas presiones de los botones en menos de 5 segundos y el documento de requerimientos establece que “el programa debe responder a todas las presiones de los botones en menos de 10 segundos”, ese requerimiento es incorrecto [Davis 1993], y difícil que se pueda detectar automáticamente.

8.1.2. No ambiguo

([Davis 1993], [IEEE 1998], [Wilson 1997])

Un documento de requerimientos es *no ambiguo* si, y solo si, cada requerimiento declarado tiene una sola interpretación. Como mínimo, esto requiere que cada característica del producto final sea descrita usando un solo término. En los casos donde el término usado en un contexto particular pudiese tener múltiples definiciones, ese término debe ser incluido en un diccionario donde su significado sea más específico [IEEE 1998].

Si se especifica por medio del lenguaje natural, esta característica es difícil de lograr ya que la ambigüedad es un atributo inherente a todos los lenguajes naturales [IEEE 1998]. Si a diez personas se les pide la interpretación de un requerimiento documentado, y se da más de una, entonces el requerimiento es ambiguo [Davis 1993], situación que puede ocurrir con una probabilidad alta.

8.1.3. Completo

([Davis 1993], [IEEE 1998], [Wilson 1997])

Un documento de requerimientos es *completo* si posee:

a) Todos los requerimientos significativos, sean estos relacionados con funcionalidad, rendimiento, restricciones de diseño, atributos o interfaces externas [IEEE 1998]. En otras palabras, lo que el software debe hacer está incluido en el documento de requerimientos [Davis 1993].

   La verificación de esto es difícil debido a la dificultad de encontrar algo que no se encuentra presente, examinando lo existente. Los únicos capaces de detectar estas omisiones son aquellos dueños del problema que debe ser resuelto por el sistema [Davis 1993]

b) La definición de todas las respuestas del software a todas las clases realizables de entrada de datos en todas las clases realizables de situaciones. Nótese que es importante especificar las respuestas a datos válidos e inválidos [IEEE 1998].

c) Desde el punto de vista de procesamiento de texto debe incluir:

   - Todas las páginas numeradas [Davis 1993].

   - Todas las etiquetas y referencias a todas las figuras, tablas y diagramas contenidas en el documento de requerimientos [IEEE 1998].

- La definición de los términos y unidades de medida [IEEE 1998].

- Todo material y secciones referenciadas deben estar presentes [Davis 1993].

d) No deben incluirse frases que impliquen algo por ser determinado en un futuro[20]. Si fuera necesario, se debe especificar la causa, qué se debe hacer para eliminarla, quién es responsable y cuándo tiene que eliminarse [IEEE 1998].

8.1.4. Verificable

([Davis 1993], [IEEE 1998], [Wilson 1997])

Un documento de requerimientos es *verificable* si, y solo si, cada requerimiento declarado es verificable. Un requerimiento es verificable si, y solo si, existe algún proceso efectivo de costo finito, con el cual una persona o máquina pueda comprobar que el producto de software cumple con el requerimiento. En general, cualquier requerimiento ambiguo no es verificable [IEEE 1998].

Requerimiento no verificables incluyen frases como “trabaja bien”, “buena interfaz usuario”, “usualmente debe ocurrir”. Estos requerimientos no puede verificarse porque es imposible definir términos “bien”, “buena” y “usualmente”. La declaración “el programa no debe entrar en un ciclo infinito”, no es posible verificarla porque la demostración automática es teóricamente imposible [IEEE 1998].

8.1.5. Consistente

([Davis 1993], [IEEE 1998], [Wilson 1997])

Un documento de requerimientos es *consistente* si, y solo si:

a) Si ningún requerimiento declarado está conflicto con otros documentos de más alto nivel, como un documento de requerimientos del sistema[21] [IEEE 1998].

b) Si no existe conflicto entre requerimientos individuales que definen las capacidades esenciales del comportamiento, y si las propiedades y restricciones del comportamiento especificado, no tienen un impacto adverso en el comportamiento [Wilson 1997].

[20] En inglés existe la frase “To be determined” o sus siglas “TBD”, que implican la definición de algo posteriormente. En español no existe una frase estandarizada, por lo que debe redactarse un término o frase que dé a entender esto, por ejemplo: “por determinar”.
[21] Sistema, en este caso, se utiliza para un problema que involucre hardware y software.

Para [Davis 1993], los conflictos entre requerimientos pueden ser de cuatro tipos:

a) Comportamiento conflictivo: Dos partes del documento de requerimientos especifican diferentes (y conflictivos) estímulos que inducen a una misma respuesta, o especifican diferentes (y conflictivas) respuestas a un estímulo idéntico y con las mismas condiciones.

b) Términos conflictivos: Dos o más términos son usados en diferentes contextos para expresar lo mismo.

c) Características conflictivas: Dos partes del documento de requerimientos demandan al producto exhibir características contradictorias.

d) Inconsistencia Temporal: Dos partes del documento de requerimientos demandan al producto a obedecer características temporales contradictorias. Por ejemplo, el documento establece que "la entrada A del sistema podrá ocurrir solo cuando la entrada B del sistema esté ocurriendo", y en otro lado establece que "la entrada B del sistema puede empezar 15 segundos después de que la entrada A del sistema haya ocurrido".

8.1.6. Comprensible por el cliente

([Davis 1993])

Pareciera que este atributo es redundante al establecer la propiedad de *no-ambigüedad* al documento de requerimientos, sin embargo, si se recurre a notaciones extremadamente formales para evitar esto, posiblemente los requerimientos expresados en esta forma no serían comprensibles para aquellos que no son especialistas en esa notación, los cuales pueden ser la mayoría de los clientes y usuarios [Davis 1993].

Una forma de evitar esto sería crear una aplicación que transforme las representaciones formales a lenguajes naturales, sin embargo, si existiera un mapeo no ambiguo a la representación informal, no tendría objetivo la expresión formal [Davis 1993].

8.1.7. Modificable

([Davis 1993], [IEEE 1998], [Wilson 1997])

Un documento de requerimientos es *modificable* si, y solo si, su estructura y estilo son tales que cualquier cambio a los requerimientos puede hacerse de forma fácil, completa y consistente, reteniendo la estructura y el estilo [IEEE 1998]. Este atributo por lo general requiere:

a) Tener una organización coherente y fácil de leer con una tabla de contenidos, índice y referencias cruzadas explícitas.

b) No ser redundante, esto es, que el requerimiento no se encuentre repetido en el documento.

c) Expresar cada requerimiento separadamente, en vez de entremezclarlos.

8.1.8. Rastreable

([Davis 1993][22], [IEEE 1998], [Wilson 1997])

Un documento de requerimientos es *rastreable* si es claro el origen de cada requerimiento y si facilita la referencia a cada requerimiento en desarrollos futuros o cuando se amplíe la documentación [IEEE 1998].

Dos tipos de rastreabilidad son recomendados:

- Rastreo *retrospectivo* (es decir, hacia etapas previas del desarrollo): Esto es donde el requerimiento referencia explícitamente su origen en documentos predecesores [IEEE 1998]. Este tipo de rastreo es importante para localizar la razón que fundamenta algún requerimiento y así determinar si es posible cambiarlo en caso necesario [Davis 1993].

- Rastreo *prospectivo* (es decir, hacia etapas posteriores del desarrollo): Esto significa que cada requerimiento debe tener un identificador único, ya sea por nombre o número. Este rastreo es importante cuando el producto entra en las fases de operación y mantenimiento. Cuando los documentos de diseño y el código son modificados, es esencial conocer el conjunto completo de requerimientos que pueden ser afectados por estas modificaciones [IEEE 1998].

8.1.9. Independiente del diseño

([Davis 1993])

Un documento de requerimientos es *independiente del diseño* si no implica una arquitectura específica del software o algún algoritmo. Cada requerimiento limita las alternativas de diseño, sin embargo, ninguno debe limitar el diseño a una sola posibilidad. Existen ciertas excepciones, principalmente para aplicaciones específicas cuyo motivo primordial es el uso de una nueva arquitectura o algoritmo [Davis 1993].

[22] [Davis 1993] considera el rastreo hacia adelante y hacia atrás como dos atributos distintos. Al rastreo hacia atrás lo denomina "rastreado" y hacia adelante "rastreable".

8.1.10. Clasificado por importancia y/o estabilidad

([Davis 1993][23], [IEEE 1998], [Wilson 1997])

Un documento de requerimiento se encuentra *clasificado por importancia y/o estabilidad*, si cada requerimiento en éste tiene un identificador que denote la importancia o estabilidad de ese requerimiento en particular [IEEE 1998].

El grado de importancia permite a los desarrolladores tomar decisiones correctas de diseño e invertir niveles apropiados de esfuerzo de acuerdo con su trascendencia. Además, permite a los clientes hacer una consideración más cautelosa de cada requerimiento, lo cual usualmente clarifica cualquier suposición escondida que tengan [IEEE 1998]. El grado de importancia puede ayudar en otros aspectos también, como el determinar el rigor y exhaustividad de las pruebas de aceptación.

El grado de estabilidad se refiere a cuán volátil es el requerimiento para guiar a la organización desarrolladora en los puntos donde debe ofrecer más flexibilidad [Davis 1993].

8.1.11. Conciso

([Davis 1993])

Dados dos documentos de requerimientos para el mismo sistema, cada uno exhibiendo niveles idénticos de las cualidades previamente mencionadas, el documento más corto es mejor [Davis 1993].

8.1.12. Organizado

([Davis 1993])

Un documento de requerimientos se encuentra *organizado* si los requerimientos son fáciles de localizar. Esto implica organizar los requerimientos de forma que los que estén relacionados se encuentren co-localizados [Davis 1993].

8.2. Clasificación de los atributos de calidad

La Figura 8-1 muestra la clasificación de los atributos de acuerdo con el modelo de calidad de [Fabbrini 1998]. En esta figura, claramente se notan factores muy débiles en cuanto criterios y otros muy fuertes. La mayoría de los atributos corresponden a validar la completitud estructural y la validez semántica. En general, estos atributos, están orientados al tipo de calidad semántica,

[23] A esta cualidad [Davis 1993] la llama “anotada” con dos tipos “necesidad relativa” y “estabilidad relativa”.

cuidando solo algunos aspectos estructurales y pragmáticos. La calidad sintáctica queda ignorada.

Tipos de Calidad	**Factores**	**Criterio**	**Explicación**
Sintáctica	Validez Sintáctica		
Estructural	Validez Estructural		
	Completitud estructural	Modificable (8.1.7)	Una estructura rígida es una debilidad del documento.
		Rastreable (8.1.8)	Para que un documento sea completo, todas la referencias deben ser válidas.
		Clasificado por importancia y/o estabilidad (8.1.10)	La prioridad debe ser una parte esencial de la estructura del documento.
Semántica	Validez Semántica	Correcto (8.1.1)	Establece que cada requerimiento que sea correcto y relevante al problema en cuestión.
		No Ambiguo (8.1.2)	Podría ser catalogado de calidad pragmática, sin embargo, un enunciado, puede ser comprendido, pero inválido debido a su ambigüedad.
		Independiente del diseño (8.1.9)	Si involucra aspectos del diseño, contiene enunciados más allá del dominio. No se cumple que $R \setminus D \neq \emptyset$
		Verificable (8.1.4)	Si un requerimiento no es verificable, por lo general, es porque es ambiguo o está mal planteado.
	Completitud Semántica	Completo (8.1.3)	Especifica que el documento de requerimientos contenga o implique cada enunciado correcto y relevante acerca del problema en cuestión.
		Consistente (8.1.5)	Si los requerimientos son completos, no pueden existir contradicciones, a menos que el dominio las posea.

Pragmático	Comprensión válida	Comprensible por el cliente (8.1.6)	Indica que todos los requerimientos hayan sido entendidos por los participantes.
		Organizado (8.1.12)	Aunque pareciera más un criterio de la calidad estructural, es un aspecto que más relacionado con la comprensión del documento.
	Comprensión completa	Conciso (8.1.11)	Un documento conciso, permite una mejor y más completa comprensión.

Figura 8-1 Atributos clasificados por tipo de calidad

La razón principal de esta falta de balance en los criterios, es por la forma como fueron desarrollados. Se trató de buscar cuáles atributos eran necesarios en un documento de requerimientos, sin importar el tipo de calidad.

Sin embargo, esto no significa que los atributos planteados no sean válidos o completos, ya que, por ejemplo, la satisfacción de algunos implícitamente provoca la satisfacción de otros no especificados. Por ejemplo, el criterio de "comprensible por el cliente", puede implicar la correctitud sintáctica y semántica.

9. Problemas de calidad en los documentos de requerimientos

En una evaluación realizada por la NASA sobre documentos de requerimientos, considerados de poca calidad, se encontraron las siguientes deficiencias [Wilson 1997]:

a) **Falta de estandarización:** Muchas veces, los estándares de documentación y estilo no eran usados. Al parecer, en algunos casos se improvisaba su estructura. En otros casos, los estándares eran aplicados de la forma incorrecta. Además, se detectó que el estándar guiaba el proceso de análisis, sin ninguna intención de adaptar el contenido al problema real. Algunas secciones tenían información no significativa debido a que la sección era parte del estándar y aparentemente el autor no deseaba dejarla en blanco.

b) **Falta de organización:** La información era organizada de una forma pobre e inconsistente. Descripciones de proyecto, el ambiente operacional y los requerimientos para la capacidad deseada eran entremezcladas. La estructura inconsistente de esta información y el uso indisciplinado de "will", "shall" y "should" [24] oscurecía la distinción entre "qué es", "qué debe ser" y "que podría ser" .

c) **Falta de entendimiento de requerimientos:** Los niveles de detalle variaban considerablemente. Aparentemente ciertos requerimientos eran bien entendidos y otros carecían de profundidad. Otra vez, parecía que los documentos de requerimientos eran usados para llenar los espacios en blanco, en vez de ser un repositorio de los resultados de un análisis de requerimientos adecuado.

d) **Estructura inconsistente:** Las secciones de los documentos, párrafos y los enunciados de los requerimientos eran identificados de forma inconsistente. Al nivel más bajo de especificación, la identificación se hacía por números o letras sin importar qué se había utilizado en los niveles superiores.

e) **Redundancia en conceptos y requerimientos:** El excesivo uso de palabras fue principalmente encontrado en los documentos grandes. En general, estos documentos intentaban combinar conceptos y requerimientos. Aparentemente, el autor recurría a aumentar la cantidad de palabras utilizadas en las descripciones en un intento por hacer al documento más interesante o esconder el hecho que había escasez de datos.

f) **Enunciados mal escritos:** Los enunciados individuales de requerimientos eran pobremente estructurados y descritos con palabras de forma pobre. Algunos enunciados eran demasiado largos, conteniendo demasiadas cláusulas compuestas y utilizaban lenguaje impreciso.

[24] Los documentos de requerimientos utilizados estaban escritos en Ingles, por lo cual el análisis se realizó en este lenguaje. Los términos se ponen tal como son, debido a que su difícil traducción a español.

Estas deficiencias abarcan todos los tipos de calidad esperados en un documento de requerimientos. Los enunciados mal escritos revelan un problema de calidad sintáctica. La falta de estandarización y la estructura inconsistente, es una falta de calidad estructural. La falta de entendimiento de requerimientos y la redundancia es un aspecto de la calidad semántica. La calidad pragmática es afectada por todos las deficiencias encontradas.

10. Estructura y organización del documento de requerimientos

El contenido del documento de requerimientos no es lo único importante para crear una especificación de calidad, también influye su estructura y organización. Esta sección manifiesta la necesidad de un estándar para lograr que los documentos muestren un estilo y estructura adecuados para cumplir con la calidad estructural. También se plantean aspectos de organización, fundamentales para lograr la calidad pragmática del documento.

10.1. Estructura del documento de requerimientos

El documento de requerimientos tiene como objetivo comunicar los requerimientos del sistema a los clientes, usuarios, gerentes y desarrolladores [Sawyer 1998], y uno de los requisitos para lograr esto es estructurarlo de una forma común y adecuada para la organización.

Esta estructura común debe ser definida como un estándar en la organización y tiene que ser verificada como parte del proceso de aseguramiento de calidad de los documentos. Debido a la diversidad de los diferentes tipos de sistemas, se pueden necesitar variantes de un mismo estándar dependiendo del tipo de sistema y la posible audiencia [Sawyer 1998]; en el estándar IEEE 830 [IEEE 1998] se sugieren varias posibles estructuras (funciones, eventos, objetos, entradas, salidas, clases de usuarios, etc.).

Para permitir esta flexibilidad es útil definir cuáles partes del estándar son variables y cuales son estáticas. Por ejemplo, la introducción y el diccionario son partes estáticas que deben aparecer en todos los documentos de requerimientos [Sawyer 1998]. En general, a pesar de ser un estándar común a una organización, éste no debe ser rígido, con el fin de incorporar o eliminar secciones de acuerdo con las necesidades específicas del problema.

El uso de una estructura común en la organización tiene los siguientes beneficios [Sawyer 1998]:

a) Un formato estándar para el documento de requerimientos significa que el lector puede usar su conocimiento previo cuando lee nuevas especificaciones. Se puede encontrar la información más fácilmente y entender las relaciones entre las diferentes partes del documento.

b) Un formato estándar actúa como una lista de verificación para los escritores de los requerimientos y reduce el riesgo de omitir información accidentalmente. De la misma forma, los lectores pueden usar el estándar para verificar si existen secciones no incluidas y como un medio para el proceso de revisión.

c) Se puede desarrollar software que soporte la producción del documento de requerimientos conforme al estándar común.

Algunos ejemplos de estándares de requerimientos son:

- Departamento de Defensa de los Estados Unidos (DoD) DI-MCCR-80025A.
- SMAP-DID-P200-SW de la NASA.
- IEEE Std 830-1998.

Los dos primeros son ejemplos de documentación de estándares típica del Gobierno de los Estados Unidos, los cuales proveen un único lineamiento estándar que debe ser usado para redactar documentos de requerimientos [Davis 1993]. El tercero es probablemente el estándar más accesible [Sawyer 1998], ofrece una buena dirección y flexibilidad concerniente a la estructura del documento [Davis 1993].

La estructura general que propone el IEEE es la siguiente [Sawyer 1998]:

1 Introducción
  1.1 Propósito del documento de requerimientos.
  1.2 Alcance del producto.
  1.3 Definiciones, acrónimos y abreviaturas.
  1.4 Referencias.
  1.5 Vistazo general del resto del documento.

2 Descripción General
  2.1 Perspectiva del producto.
  2.2 Funciones del producto.
  2.3 Características del usuario.
  2.4 Restricciones Generales.
  2.5 Supuestos y dependencias.

3 Requerimientos específicos, que cubren aspectos funcionales, no funcionales y de interfaz. Estos deben documentar interfaces externas, funcionalidad, requerimientos de rendimiento, requerimientos de la base de datos conceptual, restricciones de diseño, atributos del sistema y características de calidad.

No importa la base que se utilice para definir el estándar del documento, para que éste sea útil debe reflejar las mejores prácticas de la organización. La mejor estructura depende de las costumbres y las prácticas de la empresa, el tipo de sistema por desarrollar y los procesos para el desarrollo de sistemas [Sawyer 1998].

En general un documento de requerimientos debe incluir al menos [Sawyer 1998]:

a) Un vistazo general del sistema y los beneficios que se obtendrían si se decide desarrollar.

b) Un diccionario que explique los términos técnicos usados.

c) Una definición de los servicios o requerimientos funcionales del sistema.

d) Una definición de las propiedades del sistema o requerimientos no funcionales, como por ejemplo, confiabilidad, seguridad, etc.

e) Una definición del ambiente operativo del sistema y los cambios necesarios en ese ambiente.

f) Si se requiere una especificación detallada, se puede incluir especificaciones como modelos de sistemas mostrando las relaciones entre los componentes (esto no se refiere a diseño).

10.2. Organización del documento de requerimientos

Los métodos de documentación se deben enfocar a tratar de lograr los beneficios esperados. Muchas compañías tratan de encasillar el documento en un estándar rígido, lo cual puede provocar desinterés ya que no se acomoda a los objetivos de los lectores.

La organización es un principio fundamental para exponer las ideas de forma útil a los lectores. Sin una organización adecuada, difícilmente los usuarios del documento se referirán a éste, elevando el riesgo de desarrollar un sistema inadecuado.

El primer principio de la organización es el siguiente: “hay un lugar para cada detalle y cada detalle en su lugar”. La organización es el proceso de inventar lugares para todos lo detalles. Por lo tanto, para organizar, se deben tener detalles con necesidad de organización [Kovitz 1999].

Para lograr una organización adecuada, se debe hacer una lista desordenada de contenidos con tópicos grandes, pequeños, conceptos, proposiciones, gráficos, diagramas y cualquier otra cosa que se desea incluir en el documento. Se debe ir agregando sin pensar en el orden (secuencia) y la jerarquía en el cual aparecerán en el documento final. Cuando se va redactando cada elemento, se pueden ir haciendo agrupaciones, sin embargo, éste no debe ser el principal objetivo [Kovitz 1999].

Este estilo se enfoca en el detalle para luego ir a lo general. En principio pareciera que esta forma es la inadecuada para afrontar los requerimientos, sin embargo, si se utiliza el estilo contrario, de lo general a lo concreto, se corre el riesgo de dejar elementos necesarios por fuera, ya que no se pueden acomodar en ningún elemento. Además, no tiene sentido realizar una estructura de un documento si no se conocen los detalles.

Una vez obtenidos los elementos, se debe empezar el proceso de agrupación, el cual involucra dos aspectos fundamentales: la agrupación de la información en unidades y el orden en que las unidades son presentadas. Escoger una buena agrupación y una buena secuencia es un asunto de identificar la estructura lógica del contenido, esto es, cuáles unidades dependen de cuáles otras [Kovitz 1999].

10.2.1. Agrupación [Kovitz 1999]

Idealmente, la información que se encuentra más relacionada con otra información debe estar localizada de la misma forma físicamente y las principales divisiones en el documento deben corresponder a las divisiones principales lógicas del contenido.

Existen algunas maneras en las cuales dos o más proposiciones pueden ser cercanas lógicamente, por lo tanto pueden ser agrupadas en un mismo grupo:

Relación lógica	Ejemplo
Proposiciones A y B son acerca del mismo sujeto	Un gen consiste de una secuencia de códigos. Un gen codifica para una proteína específica.
Proposiciones A y B tiene el mismo predicado	El motor 3 cierra la válvula G Un miembro de mantenimiento puede cerrar la válvula G
Proposiciones A y B tienen sujetos en la misma clase o que son diferentes valores de la misma variable.	El puerto I/O Ox7000, bit 0, enciende el motor 3 El puerto I/O Ox7000, bit 4-7, selecciona una de las 16 velocidades del motor 3
Proposiciones A y B tiene el mismo predicado, esto es, responden a la misma pregunta o tiene una estructura paralela.	La minería de datos debe ocurrir antes del análisis de datos. El análisis de datos debe ocurrir antes de la transmisión.

Estos ejemplos son sencillos, sin embargo, las relaciones en un sistema real son más elaboradas y se convierten en una red compleja. Es imposible establecer todas las reglas para definir la agrupación de las proposiciones, sin embargo, existe un par de guías que pueden facilitar la labor. Es importante destacar que siempre debe usarse el principio del sentido común antes que una regla.

Describir una cosa a la vez

La descripción de los requerimientos debe ser comprensible para los lectores, pero a menudo, esto no se logra al tratar de describir más de un requerimiento a la vez. Tratar de describir dos cosas a la vez, casi garantiza la incomprensibilidad. La razón del por qué esto ocurre, es debido a que la información en un documento técnico está muy interrelacionada en muchas maneras. Por ejemplo, se puede tener el temor de describir la consulta A, sin tomar en cuenta las capacidades de autorización; se piensa que el usuario puede ejecutar la consulta A incondicionalmente.

Al colocar la información del ejemplo anterior en secciones separadas se corre el riesgo que el usuario del documento leerá la sección de las consultas, pero no la de autorizaciones. Sin embargo, este riesgo es sobrellevado por otros riesgos más altos que provocados por la redundancia o por tratar de decir más de una cosa a la vez. En realidad el riesgo de confundir al lector es menor si el documento es claro, conciso e incluye nada más el contenido útil.

Siete, más menos dos

Usualmente cuando se necesita realizar organizaciones de elementos, se dice que el ideal es agrupar en conjuntos entre cinco y nueve, siendo siete el número ideal. Esta concepción se originó del artículo "The Magical Number Seven Plus or Minus Two" ("El numero mágico siete más o menos dos", de George A. Miller, en 1956. Este artículo trata de explicar las limitaciones en nuestra capacidad para procesar información.

Entre los aspectos que trata el artículo es lo referente a cuánta información se puede mantener en la memoria a corto plazo, dando cierta información a alguien y pidiéndole que la repita. La mayoría de la gente, después de escuchar una secuencia de dígitos binarios, solo pueden repetir correctamente hasta nueve dígitos.

Esto hace pensar que la capacidad se limita a nueve bits de información, sin embargo, si al escuchar cada cinco dígitos, se convierte el número a decimal y se olvida de los binarios, podría repetir más de los nueve dígitos binarios. En un binario de 40 bits de largo, esto significaría que bastaría repetir 8 números decimales.

A esto se le llama recodificación, con lo cual se aumenta nuestra capacidad de retención. Entonces, la capacidad de la memoria es medida en trozos de información, no en bits y a través de recodificación estos trozos pueden significar almacenar una gran cantidad de bits.

El número de trozos de información que se pueden retener depende del tipo de información; para bits, puede ser hasta nueve, pero a menudo es 7 o inclusive cinco para una lista de palabras arbitrarias.

La recodificación va más allá de poder memorizar más bits, mediante esta técnica nuestro cerebro expande la capacidad de memoria inmediata, permitiéndonos considerar problemas de gran complejidad, por ejemplo, leer un libro de 400 páginas sin perderse: cuando se termina un capítulo, éste se puede relegar en la mente, junto con todos sus detalles, y pensar en el siguiente capítulo. Si hay dos que están conectados de alguna forma, se puede reabrir mentalmente el capítulo, a como sea necesario, para entender las consecuencias de la conexión y luego olvidarlo de nuevo.

Por ejemplo, si se tienen 700 elementos que pertenecen a una tabla, se deben poner los 700 elementos en la tabla, no es necesario dividir la tabla. Solamente se deben considerar aspectos de presentación para hacer más fácil la lectura. El lector hará un solo bloque de memoria.

El principio original de recodificacion del artículo de Miller es el siguiente: Hacer fácil a los lectores recodificar o armar en bloques la información, para que ellos nunca tengan que considerar más de cuatro o cinco cosas a la vez (no nueve).

De hecho, se puede organizar casi toda la documentación, de forma que el lector no tenga la necesidad de pensar en una o dos cosas a la vez.

10.2.2. Secuencia [Kovitz 1999].

La secuencia ideal consiste en presentar la información de forma que ningún enunciado aparezca antes de otro que lógicamente lo requiere; sin embargo, lograr esto es muy difícil. De hecho, pareciera que en software es más complicado, ya que todas las piezas están muy integradas.

Al redactar un documento, especialmente de requerimientos, es fácil encontrarse con referencias cruzadas, es decir, que no se pueda establecer una dependencia unidireccional entre los enunciados. En estos casos, se debe buscar un orden lógico principal, es decir, lo más fundamental es lo que se debe colocar primero.

Siempre la clave para resolver este problema es poner lo que es más fundamental lógicamente de primero. La regla para determinar esto es la siguiente:

- Si el concepto o proposición B se refiere a otro concepto o proposición A, entonces A es más es más fundamental y debe colocarse primero.

- Una proposición se refiere a otra proposición mediante la extensión, variación, referencia o inclusión de ésta.

La siguiente es una lista de heurísticas para determinar cuál de entre dos piezas de información es lógicamente más fundamental:

- Los hechos de los cuales no se puede escoger, son más fundamentales que los hechos de los cuales se puede escoger. Esto es, que los enunciados descriptivos deben estar antes que los requerimientos o, en otras palabras, el problema es más fundamental que la solución.

- Las cosas son más fundamentales que sus atributos, relaciones y las acciones que pueden hacer o que pueden afectarlas.
  - Corolario: Las relaciones son menos fundamentales de los elementos que relacionan. Esto incluye las relaciones entre acciones, como relaciones causales.
  - Corolario: Las acciones son menos fundamentales que lo que cambia durante la acción.
  - Corolario: Los atributos son menos fundamentales que aquello de lo que son atributos. Esto incluye atributos de relaciones y acciones, e inclusive otros atributos.

- El caso normal es más fundamental que el caso excepcional. Los casos excepcionales son variaciones o extensiones de los casos normales.

- Una cosa es más fundamental que cualquiera de las funciones que desempeña (o como se usa) en diferentes situaciones.

- Una descripción de los agentes usualmente debe anteceder una descripción de los objetos pasivos sobre los cuales éstos actúan[25].

Estas heurísticas no son perfectas y en ciertos momentos pueden provocar contradicciones. Lo importante es que son una guía para mejorar la secuencia, sin embargo, siempre existirán interrelaciones.

Una forma de minimizar el impacto de las relaciones es colocar un resumen, al principio del documento o capítulo, con lo cual se puede lograr que el lector interprete de forma general estas relaciones sin complicarlo con detalles. Este resumen debe cumplir los mismos principios de secuencia.

10.2.3. Énfasis [Kovitz 1999].

Una de las técnicas utilizadas para obligar al lector a poner especial cuidado en un punto especifico es el énfasis. Éste tiene el objetivo de distinguir el dos por ciento del contenido que es más importante que el noventa y ocho por ciento restante.

[25] Para las técnicas de análisis orientado a objetos favorecen las descripciones basadas en *casos de uso* [Trejos 1999] es conveniente, entonces, identificar y describir primero los *actores* que interactuarán con el sistema.

La importancia es principalmente la localización de la proposición en la estructura lógica del documento. Un enunciado importante es aquel que debe ser recordado.

El énfasis es una forma de prevenir que el lector pierda atención de enunciados que son importantes para entender el resto de la información. En el documento, esto se refleja porque se da a los enunciados una importancia visual que calza con su importancia lógica. Algunas técnicas para lograr esto son (ordenadas de mayor a menor importancia):

- Un gráfico enfatiza su contenido
- Lo que aparezca primero es enfatizado.
- A lo que se refiere muchas veces, es enfatizado. Siempre y cuando cada vez se le agregue información, no basta con hacer una simple referencia.
- Rodear el contenido con espacios en blanco.
- Las viñetas enfatizan.
- La repetición es otra forma de enfatizar, por ejemplo, dando un ejemplo o mostrando un gráfico.
- Tomar más espacio enfatiza. Una sección larga parece más importante que una corta. Es importante mencionar que no es conveniente hacer la sección más larga solo con el afán de provocar énfasis. Un tópico importante por sí solo tendría una sección más larga.
- Cualquier clase de contraste enfatiza el elemento contrastado.
- El uso de cursiva (itálica) es el último recurso ya que es la técnica más débil.

11. Validación y verificación

Es sabido que el costo de reparar un error es más bajo si se corrige cerca de la fuente que, cuando se está más avanzado en el desarrollo. En el caso de requerimientos, Wiegers cita estudios que indican que cuesta entre 68 y 110 veces más corregir un defecto de requerimientos encontrado por el cliente (o donde él) que durante el proceso de especificación de estos [Wiegers 1999].

Por esta razón, en todas las etapas del desarrollo de sistemas se incluyen, o se sugiere incluir, actividades de validación y verificación, las cuales tienen como objetivo final detectar las fallas lo más cerca posible de su origen.

En la práctica, no suele hacerse una distinción en el uso de los términos de verificación y validación. Sin embargo, sí tienen una diferencia, de acuerdo con el objetivo que buscan:

Validación: Evaluar un producto de software entregado para determinar si funciona de acuerdo con lo que prescriben los requerimientos [Trejos 1997].

Verificación: Determinar si cada actividad del ciclo de vida del proyecto interpreta correctamente la especificación dada por la actividad precedente y genera un producto que satisface esa especificación [Trejos 1997].

La diferencia entre estos dos términos es que la verificación está orientada a revisar que las cosas se estén haciendo de la forma correcta. Validación es el proceso para determinar si se está haciendo la cosa correcta [Wiegers 1999].

En el caso de la etapa de análisis, la diferencia entre ambos términos se vuelve más sutil, incluso hay autores que eligen uno para describir el proceso y otros lo utilizan de forma indistinta. Sin embargo, es importante definir claramente qué significa cada uno para el proceso de requerimientos:

Validación: Es la parte del proceso orientada a revisar que los requerimientos especifiquen lo que el cliente realmente quiere (que se esté especificando lo correcto).

Verificación: Es la parte del proceso que asegura que los requerimientos cumplan con las características de calidad.

Muchas veces el proceso de verificación y validación de requerimientos es omitido, porque los participantes consideran que incluir esta etapa en el cronograma, afecta la fecha de entrega del producto por el mismo tiempo destinado a esa actividad. Esta percepción supone que no existe retorno al invertir tiempo en el proceso de verificación y validación, pero, por el contrario, esta inversión puede reducir el tiempo de desarrollo mediante la disminución del retrabajo y

acelerando las pruebas del sistema. Carpers Jones reporta que cada dólar gastado en prevención de defectos reduce la reparación de defectos de tres a diez dólares [Wiegers 1999].

Además, mejores requerimientos conllevan mayor calidad del producto y mayor satisfacción del cliente, lo cual reduce el costo de la vida del producto para el mantenimiento, mejoras y soporte. Invertir en la calidad de los requerimientos casi siempre ahorra más dinero de lo gastado [Wiegers 1999].

11.1. Revisiones estáticas

Para llevar a cabo la validación de los requerimientos, es necesario recurrir a las revisiones estáticas las cuales, a diferencia de otros tipos de comprobación, no requieren el producto de software terminado para llevarlas a cabo.

Una revisión estática es una actividad grupal en la cual se revisa un producto del proceso de desarrollo con el propósito de encontrar problemas y mejorar su calidad. (Se denominan estáticas porque los productos no se someten a procesos de ejecución computacional.) [Trejos 1997]

Todas las modalidades de revisión estática se basan en el principio básico de *someter los productos (documentos) de un proceso de desarrollo de sistemas a la revisión por parte de personas (revisores) distintas de los productores (autores)* [Trejos 1997].

Las revisiones estáticas son la herramienta adecuada para el proceso de validación de la etapa de análisis, ya que se pueden aplicar desde etapas tempranas del desarrollo de sistemas, sobre cualquier producto del proceso y tienen una efectividad muy superior (encuentran del 60% al 90% de los defectos) [Trejos 1997].

11.1.1. Variedades de revisiones estáticas [Trejos 1997]

Esta es una lista general de las variedades de revisiones estáticas:.

a) *Vistazos generales*: Proveen información acerca de productos y procesos. Permiten capacitar a nuevos miembros de equipos. Sirven para preparar inspecciones, revisiones técnicas o recorridos (“walkthroughs”).

b) *Revisiones técnicas*: Se usan en combinación con las inspecciones y los recorridos. Se usan principalmente para la identificación y resolución de asuntos técnicos relativos a la especificación o diseño de elementos del software.

c) *Inspecciones*: Las inspecciones son la variedad de revisión estática más formal y están enfocadas en la detección y clasificación de problemas. Aseguran que el producto conforma a su especificación, es correcto y cumple con los estándares requeridos.

d) *Revisiones administrativas*: Dan elementos para tomar decisiones sobre acciones futuras, con base en el desempeño del proyecto. Esto comprende: hacer que las actividades avancen de acuerdo con el plan (con base en evaluaciones del estado del desarrollo de productos), cambiar la dirección del proyecto, identificar revisiones a los planes, reasignar recursos.

e) *Revisiones informales*: Están orientadas a los propósitos del desarrollador.
   - Los recorridos evalúan un elemento de software con el propósito de: encontrar defectos, omisiones o contradicciones; mejorar el elemento; considerar implementaciones alternativas.
   - Las tormentas de ideas evalúan generalmente grandes alternativas de diseño, elecciones tecnológicas de amplio impacto, elementos de riesgo, etc.

f) *Auditorías*: Proveen confirmación independiente y objetiva de los productos y procesos para certificar el cumplimiento de estándares, lineamientos y especificaciones. También pueden verificar que los productos no tienen características adicionales no solicitadas o no deseadas por los usuarios. Las auditorías siguen planes específicos, sus resultados son documentados sistemáticamente y entregados a los entes correspondientes (opcionalmente con recomendaciones de revisión y acciones subsecuentes).

Para efectos de la etapa de análisis, los recorridos y las inspecciones son las técnicas más adecuadas para realizar la revisión del documento de requerimientos. Los recorridos o "walkthroughs" son más informales y pueden realizarse aún cuando el documento no esté totalmente terminado. Las inspecciones son formales y rigurosas; se deben realizar antes de continuar con la siguiente etapa del desarrollo.

Las inspecciones requieren un nivel de madurez avanzado en gestión de calidad en la organización, sin embargo, esto no evita que sean adaptables y aplicables. Los recorridos son una técnica informal o, por lo menos, el nivel de formalidad lo establece el organizador. Lo ideal consiste en desarrollar los requerimientos aplicando ambas herramientas, ya que atacan problemas distintos. Los recorridos tratan de prevenir defectos durante el establecimiento de los requerimientos y las inspecciones son un punto de control antes de seguir con la siguiente etapa.

Ambas técnicas pueden adaptarse según las necesidades de cada proceso de desarrollo, y el nivel de madurez de la empresa en el tema de calidad. Sin embargo, es importante considerar el riesgo que conlleva atenuar la aplicación de estas prácticas, en función del costo por re-trabajo.

11.2. Inspección [Wiegers 1999]

Michael Fagan desarrolló el proceso de inspección en IBM a medidos de los años setenta y ha sido reconocido como una buena práctica en la industria del software. Cualquier producto de software puede ser inspeccionado, incluyendo los documentos de requerimientos y diseño, código fuente, documentación de pruebas, planes de proyectos, etc.

La inspección es un proceso bien definido de múltiples etapas que involucra un pequeño equipo de participantes entrenados que se enfocan en pedazos del producto de trabajo para encontrar defectos. Las inspecciones proveen un filtro de calidad a través del cual los documentos deben pasar para ser aceptados para la siguiente fase.

11.2.1. Participantes

Los participantes en el proceso de inspección deben representar tres puntos de vistas:

1. El autor del producto o representantes del autor: El analista que escribió el documento de requerimientos provee este punto de vista.
2. El autor de cualquier producto predecesor o especificación para el ítem que se inspeccionará. Puede ser el ingeniero o arquitecto del sistema, el cual puede examinar el documento para rastrear cada requerimiento a una especificación del sistema. Si no existe un nivel más alto de requerimientos, la inspección puede incluir a los clientes para asegurar que el documento describe el problema de forma completa y exacta.
3. Las personas que deben realizar trabajo basándose en el documento que se está inspeccionando. Se puede incluir a un desarrollador, un miembro del equipo de pruebas, un director de proyectos y un miembro del equipo de documentación. Estos detectarán diferentes clases de problemas de acuerdo con su punto de vista.

Los equipos no deben ser mayores a siete personas. Los equipos grandes tienen a desenfocarse del problema principal.

11.2.2. Papeles en la inspección [Wiegers 1999] [Trejos 1997]

Durante la inspección los participantes asumen un papel específico, los cuales pueden variar de una inspección a otra, pero las funciones que realizan son las mismas.

a) *Autor*: El autor que creó o mantiene el producto inspeccionado. En el caso de requerimientos, usualmente el analista que recolectó los requerimientos del cliente es quien escribió el documento. Debe asumir un papel pasivo y atender (no debatir) preguntas especificas que el lector es incapaz de responder. No debe ser moderador, lector o escriba. Es probable que encuentre errores que los otros inspectores no detecten debido a su conocimientos.

b) *Moderador*: El moderador o líder de inspección, planea la inspección con el autor, coordina actividades y facilita las reuniones de inspección. El moderador distribuye los materiales a ser inspeccionados por los participantes varios días antes de la reunión, empieza las reuniones a tiempo, promueve la participación y se encarga de enfocar la reunión en encontrar errores más que en resolver los errores que surjan. También, el moderador debe reportar los resultados a la gerencia o encargado de recolectar los datos de las inspecciones. Además, debe darle seguimiento con el autor a los cambios propuestos, para asegurarle que los defectos y las situaciones encontradas sean resueltas de la forma apropiada.

c) *Lector*: Dirige al equipo a través del producto revisado, de manera lógica y completa, durante la inspección. Parafrasea el material y describe sus componentes. En el caso de requerimientos, el lector debe presentar un requerimiento o un párrafo a la vez.

d) *Escriba*: El escriba se encarga de documentar, en formularios estándar, los aspectos y defectos encontrados durante la inspección. El lector debe leer lo escrito para confirmar que lo documentado sea preciso. Además, asiste al moderador en la preparación de los informes de inspección.

e) *Inspectores*: Todos los miembros del equipo son inspectores, los cuales deben detectar defectos, pero pueden ofrecer soluciones[26]. Deben tener presente que todos revisan el producto y no quién lo produjo.

[26] Dado que las inspecciones se propusieron originalmente para revisar código, en la literatura se recomienda que los inspectores se enfoquen en encontrar defectos y no en sugerir soluciones. Sin embargo, en el caso de inspeccionar requerimientos, es fundamental que se aclare y mejore todo lo posible en el documento de especificación de requerimientos.

11.2.3. Etapas del proceso de inspección [Wiegers 1999]

El proceso de inspección se resume la Figura 11-1.

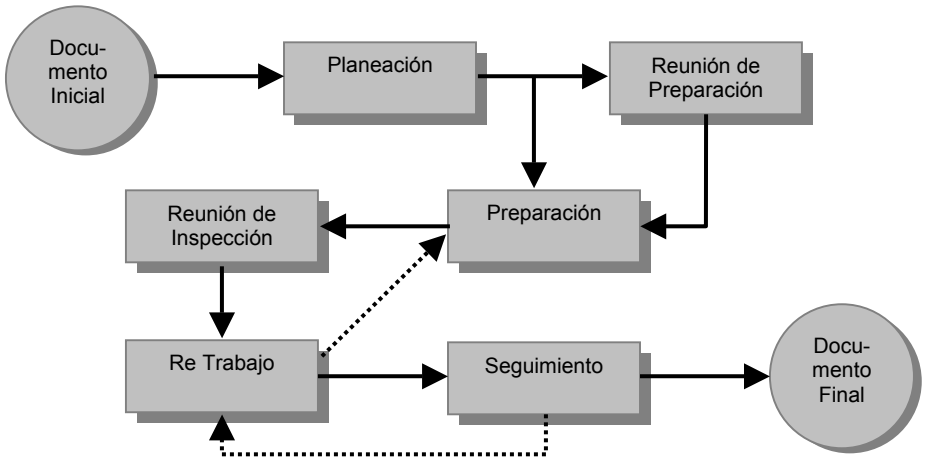


Figura 11-1 Proceso de inspección

Planeación: El autor y el moderador planean de forma conjunta la inspección, determinando quiénes deben participar, cuáles materiales deben recibir los inspectores antes de cada reunión y cuántas reuniones de inspección serán necesarias.

La cantidad de reuniones puede ser calculada de acuerdo con la tasa de inspección, en unidades como páginas por hora. La elección de esta tasa tiene una incidencia alta en la cantidad de defectos detectados, como se muestra el gráfico de la Figura 11-2 .

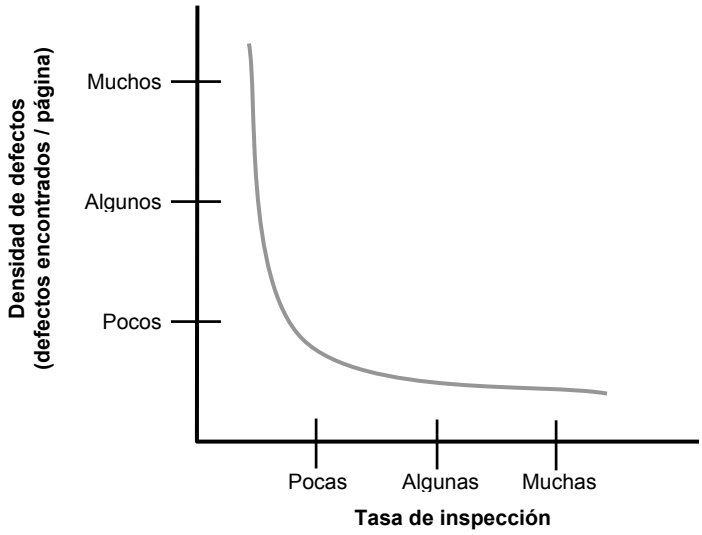


Figura 11-2 Densidad de defectos por tasa de inspección

Si la inspección es más pausada, se pueden detectar más defectos, sin embargo, el tiempo que consume la inspección puede ser alto y no siempre todos los participantes disponen de ese tiempo. Debido a esto, es necesario escoger una tasa que se adecue a los tiempos disponibles, considerando el riesgo de pasar por alto defectos grandes. Una tasa de seis páginas por hora es una tasa aceptable, la cual se puede ajustar de acuerdo con los siguientes factores:

- Cantidad del texto en la página
- La complejidad de la especificación
- Qué tan crítico es el material para el éxito del proyecto
- Datos de inspecciones previas
- Nivel de experiencia del autor del documento de requerimientos.

Reunión de preparación: Esta reunión provee una oportunidad para informar a los inspectores acerca de los antecedentes del material que inspeccionarán, cualquier supuesto realizado por el autor y los objetivos específicos del autor. La reunión se puede omitir si todos los inspectores están familiarizados con los elementos por inspeccionar.

Preparación: Durante la preparación de una inspección formal, cada inspector, utilizando una lista de cotejo de defectos típicos como guía, examina el producto identificando posibles defectos y cuestionamientos. Hasta un 75 por ciento de los defectos encontrados por la inspección, son detectados en esta etapa. Si no existe una preparación adecuada, las reuniones resultan inefectivas, creando la percepción de que las inspecciones son una pérdida de tiempo.

Reunión de inspección: Durante la reunión de inspección el lector lleva al equipo a través del documento de requerimientos, parafraseando un requerimiento a la vez. Cuando se encuentran defectos y otros aspectos, el escriba los registra en un documento que se convertirá en la lista de acciones que debe realizar el autor de los requerimientos.

El propósito de la reunión es identificar tantos defectos u oportunidades de mejora en el documento de requerimientos como sea posible. Cualquier otro aspecto, como de forma, alcance del proyecto, posibles soluciones, etc., es útil pero pueden desviar el objetivo de la reunión.

La reunión de inspección no debe tardar más de dos horas, de lo contrario es necesario programar una nueva reunión. Al final de la reunión, el equipo debe decidir si los documentos de requerimientos son aceptados, aceptados con cambios menores o no aceptados porque se debe hacer una revisión mayor y una reinspección.

Retrabajo: Casi toda actividad de control de calidad siempre provoca un retrabajo, por lo cual, el autor debe planear tiempo para esta actividad después de la reunión de inspección. Es muy costoso corregir los requerimientos durante la siguiente etapa; es mejor hacer una pausa para llevar a cabo la corrección de esos defectos.

Seguimiento: En esta etapa final, el moderador o un inspector previamente designado, realiza un seguimiento en el retrabajo que el autor ha llevado a cabo. En el seguimiento se asegura que todos los aspectos han sido atendidos y que los requerimientos fueron corregidos de la forma correcta.

11.2.4. Criterios de entrada y salida [Wiegers 1999].

Es posible llevar a cabo una inspección del documento de requerimientos cuando satisface una serie de prerrequisitos, los cuales deben estar documentados en forma de lista de cotejo. Estos criterios de entrada establecen expectativas claras para que el autor prepare una inspección. También evitan que se pierda tiempo en la reunión en aspectos que deben ser resueltos previo a la inspección.

La siguiente es una lista sugerida de prerrequisitos para el proceso de inspección:

- El documento está conforme a la plantilla estándar.
- La ortografía y, de ser posible, la gramática, han sido revisadas y corregidas.
- El autor ha examinado visualmente el documento por cualquier error de presentación.
- Están disponibles los documentos predecesores o de referencia que se necesitan para llevar a cabo la inspección
- Los números de línea están impresos en el documento para facilitar la referencia a lugares específicos.
- Todos los aspectos no atendidos están marcados “Por Determinar”.

- El documento incluye un glosario de términos.

De forma parecida, se deben establecer los criterios de salida, los cuales deben ser satisfechos antes de declarar la inspección como completa. La siguiente es un lista sugerida de criterios de salida:

- Todos los aspectos encontrados en la inspección han sido atendidos.
- Los cambios en el documento fueron realizados de la forma correcta.
- La ortografía y, de ser posible, la gramática, han sido revisadas y corregidas.
- Todos los puntos "por determinar", han sido resueltos o se ha establecido cuál es el proceso de resolución, fecha de entrega y encargado.
- El documento ha sido incluido en el sistema de administración de la configuración.
- Las métricas de inspección han sido reportadas al punto apropiado.

11.2.5. Lista de cotejo para inspección [Wiegers 1999].

Para ayudar a los inspectores a encontrar errores típicos en los productos que inspeccionan, es necesario crear una lista de cotejo para cada tipo de documento de requerimientos que existe en la compañía. Estas listas llaman la atención de los inspectores en aspectos donde típicamente se falla.

Las listas pueden llegar a ser muy grandes e inclusive difíciles de manejar para los inspectores. Sin embargo, no necesariamente todos deben cotejar la misma lista. Es una buena estrategia tener listas con objetivos o enfoques distintos para distribuirlas entre los participantes. Esto facilita la detección de errores, ya que cada inspector estaría enfocado a encontrar una clase de errores. A continuación aparece una lista de cotejo para la inspección.

A. Organización y Completitud

- ¿Están correctas todas las referencias cruzadas a otros requerimientos?
- ¿Todos los requerimientos fueron escritos con un nivel de detalle consistente y apropiado?
- ¿Todos los requerimientos proveen una base adecuada para el diseño?
- ¿Está incluida la prioridad de implementación para cada requerimiento?
- ¿Están definidas todas las interfaces de hardware, software y comunicaciones?
- ¿Están definidos todos los algoritmos intrínsecos a los requerimientos funcionales?
- ¿El documento incluye todos las necesidades del cliente y del sistema?
- ¿Falta alguna información necesaria para un requerimiento? ¿Si es así, está marcado "por determinar"?
- ¿Está documentado el comportamiento esperado para todos los errores previstos?

B. Correctitud

- ¿Existen requerimientos conflictivos o duplicados?
- ¿Cada requerimiento está escrito de forma clara, concisa y con un lenguaje no ambiguo?
- ¿Cada requerimiento es verificable por pruebas, demostraciones, revisiones o análisis?
- ¿Cada requerimiento está en el alcance del proyecto?
- ¿Cada requerimiento está libre de errores de contenido y gramaticales?
- ¿Pueden todos los requerimientos ser implementados con las restricciones conocidas?
- ¿Son los errores especificados de forma única y con sentido?

C. Atributos de calidad

- ¿Están todos los objetivos de rendimiento especificados de forma apropiada?
- ¿Están especificadas apropiadamente todas las consideraciones de seguridad y privacidad?
- ¿Están explícitamente documentadas y cuantificadas todas las metas de los atributos de calidad, junto con los inconvenientes y beneficios definidos?

D. Rastreabilidad ("trazabilidad")

- ¿Cada requerimiento es identificado de forma única y correcta?
- ¿Cada requerimiento funcional está ligado a un requerimiento de mayor nivel?

E. Aspectos especiales

- ¿Todos los requerimientos son realmente requerimientos, no soluciones de diseño e implementación?
- ¿Están identificadas las funciones de tiempo crítico, junto con su criterio?
- ¿Están resueltos los aspectos relacionados con la internacionalización?

11.2.6. Probar los requerimientos [Wiegers 1999]

Es difícil visualizar cómo un sistema se comportará en circunstancias especificas leyendo el documento de requerimientos. Los casos de pruebas derivados de los requerimientos pueden ayudar a hacer el comportamiento del sistema tangible a los participantes del proyecto. El simple hecho de diseñar los casos de prueba, puede revelar defectos en los requerimientos, sin necesidad de ejecutar las pruebas en sistemas operacionales.

Escribir una prueba funcional o de caja negra, plasma de alguna forma la visión de cómo debe funcionar el sistema en condiciones especificas. Los requerimientos vagos o ambiguos se mostrarán en estas pruebas, porque no se sabrá cómo describir el comportamiento del sistema.

Está técnica se puede usar junto con las inspecciones para lograr asegurar aún mas la calidad de los requerimientos. Además, sirve para que los participantes del proyecto entiendan mejor el sistema y que los clientes puedan determinar, de mejor forma, si el sistema cumplirá con las necesidades que tienen.

11.3. Recorridos o "walkthroughs" [Trejos 1997]

Los recorridos son muy semejantes a las inspecciones, con la diferencia de ser menos rigurosos. Su grado de formalidad es muy variable y depende del objetivo que se establezca para cada recorrido en particular.

A diferencia de las inspecciones, esta técnica permite que el autor pueda dirigir el proceso y revisar el documento o un conjunto de requerimientos, sin necesidad de haber concluido el proceso del desarrollo de éstos.

Los objetivos de un recorrido son:

- Evaluar el documento o un conjunto de requerimientos, en alguno o varios de los siguientes aspectos:
  - Encontrar defectos
  - Detectar omisiones
  - Encontrar contradicciones
  - Mejorar el elemento
  - Considerar descripciones alternativas
  - Determinar la implementabilidad de los requerimientos
- Intercambiar estilos y técnicas
- Informar a los participantes
- Encontrar los problemas de presentación y estructura
- Brindar un mecanismo de retroalimentación para los autores, que les permita mejorar los productos de su trabajo.

11.3.1. Papeles de los participantes

a) *Moderador*: Dirige un recorrido específico, manejando sus asuntos administrativos y asegurando que se realiza de manera ordenada. También es responsable de enunciar los objetivos.

b) *Autor*: Es el autor del producto que se está revisando. Presenta los materiales y aclara dudas. Resuelve, fuera de la reunión, los problemas encontrados.

c) *Escriba*: Registra todos los comentarios realizados durante el recorrido y los clasifica como: errores detectados, omisiones, contradicciones, sugerencias para mejoras, enfoques alternativos o cuestiones de estilo.

d) *Revisores*: Además de los anteriores, pueden haber otros participantes. Todos deben prepararse antes de la reunión y aportar al máximo para cumplir con los objetivos del recorrido.

En los recorridos es normal que el autor desempeñe el papel de moderador. El número de participantes en este tipo de revisión debe limitarse a 5 personas. El equipo mínimo está constituido por el autor (quien, a su vez, realiza el papel de moderador) y por otro revisor que asume el papel de escriba.

11.3.2. Etapas del proceso

El proceso de los recorridos es similar al proceso de inspecciones, de hecho, contiene las mismas etapas y en el mismo orden (Figura 11-1):

Planificación: El moderador verifica los criterios de entrada, establece el calendario, identifica a los revisores y les distribuye el material.

Reunión de preparación: El autor presenta el trabajo a los revisores.

Preparación: Cada revisor estudia individualmente los materiales distribuidos.

Reunión: El autor presenta brevemente el elemento de revisión y recorre los elementos. Los revisores plantean las preguntas, problemas detectados y alternativas posibles. El escriba registra los comentarios y decisiones en el informe del recorrido.

Re-trabajo: El autor revisa el producto, atendiendo a los aspectos encontrados.

Seguimiento: El moderador comprueba la resolución de los asuntos por el autor y se determina si se necesita otra revisión, o si el elemento se aprueba.

12. Conclusiones

Los requerimientos son los pilares del desarrollo de los sistemas de información. Si la calidad de los requerimientos no es la adecuada, el producto se puede afectar en tres aspectos importantes:

a) *Calidad de producto final*: Si no se tienen requerimientos adecuados (que cumplan con los atributos de calidad), es poco probable poder lograr un producto de calidad, es decir, que satisfaga las necesidades del cliente. Además, para el desarrollador se vuelve imposible demostrar la calidad del producto.

b) *Estimación*: La base para estimar son los requerimientos, si estos son especificados de una forma vaga, incorrecta o son omitidos, no es posible realizar una estimación adecuada del desarrollo en cuanto a las actividades, tiempo de duración y costos. Por lo general, la escasez de especificación de requerimientos hace que el dimensionamiento del sistema sea menor a lo real, no solo en tiempos, costos y actividades, sino también en expectativas.

c) *Costo del "retrabajo"*: La mala definición y administración de los requerimientos generan un costo adicional al proyecto durante su desarrollo, lo cual puede provocar una diferencia grande entre los tiempos y costos estimados y los reales. El costo de una modificación en los requerimientos es mucho mayor conforme se avanza en las etapas. El objetivo de una buena administración es reducir el riesgo asociado a la no adecuada definición de los requerimientos y al cambio de estos durante el desarrollo.

Acerca de los requerimientos y su importancia, [Davis 1993] concluye lo siguiente:

a) Existen muchos errores de requerimientos.

b) Muchos de estos errores pasan sin detectar.

c) Muchos de estos errores pueden ser detectados cerca de la fuente.

d) No detectar estos errores puede contribuir a incrementar los costos de software.

En cuanto a la mala administración de los requerimientos [Davis 1993] determina las siguientes consecuencias:

a) El software resultante puede no satisfacer los requerimientos del usuario.

b) Interpretaciones distintas de los requerimientos pueden causar la falta de comprensión entre usuarios y desarrolladores, gastando tiempo y dinero en demandas.

c) Es imposible probar que el software cumple con los requerimientos pretendidos.

d) Puede malgastarse tiempo y dinero en la construcción de un sistema malo.

Para minimizar estos riesgos es necesario buscar que el documento de requerimientos cumpla en todo momento con los atributos de calidad. Sin embargo, es importante notar que satisfacer todos estos simultáneamente es una labor casi imposible. Por ejemplo, si se intenta eliminar las inconsistencias y ambigüedades (usualmente eliminando el lenguaje natural), puede que el documento de requerimientos se convierta en menos comprensible para los clientes, los cuales no son especialistas en computación. Si se intenta ser absolutamente completo, los costos del documento se pueden elevar mucho, y el documento se vuelve grande y difícil de leer. Si se trata de incrementar la modificabilidad eliminando redundancia, el documento se vuelve "cortado", difícil de leer y ambiguo.

La única conclusión a la que se puede llegar es que no existe un documento de requerimientos perfecto [Davis 1993].

Bibliografía anotada

[Davis 1993]

Davis, Alan M. *Software requirements: objects, functions and states.* Prentice-Hall, Inc, EEUU, 1993.

En este libro se discute la mayoría de los aspectos relacionados con los requerimientos del software. Es muy completo y además incluye una bibliografía anotada con más de 700 referencias.

[Fabbrini 1998]

F. Fabbrini, M. Fusani, V. Gervasi, S. Gnesi, y S. Ruggieri. *Achieving Quality in Natural Language Requirements.* Proceedings of the 11th International Software Quality Week, 1998.

En este artículo se hace una propuesta de un modelo de calidad para los documentos de requerimientos. De este modelo se derivan cuatro tipos de calidad: la sintáctica, la estructural, la semántica y la pragmática.

[Graham 1998]

Graham, Ian. *Requirements engineering and rapid development: a rigorous, object-oriented approach.* Addison Wesley Longman Limited, Inglaterra, 1998.

Su enfoque primordial es hacia las técnicas orientadas a objetos para la definición de requerimientos y el desarrollo rápido de aplicaciones. Este libro trata de hacer referencia a la mayoría de los métodos orientados a objetos.

[IEEE 1998]

IEEE Std 830-1998. *IEEE Recommended Practice for Software Requirements Specifications.* IEEE, 1998. (Revisión del IEEE Std 830-1993.)

Estándar que establece cómo debe estructurarse un documento de requerimientos y cuáles son los atributos de calidad que este debe poseer para considerarlo adecuado.

[Jones 1996a]

Jones, Capers. *Conflict and Litigation Between Software Clients and Developers*. Software Productivity Research, Inc. EEUU, 1996. Tomado de Intenet, dirección: http://www.spr.com.

En este artículo se describen las causas de los conflictos entre los desarrolladores de software y sus clientes. Además, se presenta una guía para minimizar el riesgo de que estos conflictos ocurran. El autor sugiere lo siguiente:

1. Determinar el tamaño de las aplicaciones antes de firmar el contrato (por ejemplo, con puntos de función).
2. Estimación formal de costos.
3. Controlar requerimientos cambiantes.
4. Asesoría independiente para evaluación.
5. Incluir requerimientos de calidad en los contratos.
6. Establecer métodos para lograr alta calidad y excelencia.

[Jones 1996b]

Jones, Capers. *Applied software measurement: assuring productivity and quality.* McGraw-Hill, EEUU, 1996.

Este libro está enfocado a mostrar las métricas recolectadas por el autor sobre el desarrollo de proyectos informáticos. Además, da explicaciones de cómo pueden aplicarse en el desarrollo de sistemas.

[Kovitz 1999]

Kovitz, Benjamín L. *Practical Software Requirements: A manual of content and style.* Manning Publications Co., EEUU, 1999

Este libro está orientado a describir qué debe contener el documento de requerimientos y la especificación de diseño. También establece la mejor forma de obtener la información del problema y cuál es la presentación más adecuada para los requerimientos en el documento.

[McConnell 1996]

McConnell, Steve. *Rapid development: taming wild software schedules*. Microsoft Press, EEUU, 1996.

Este libro se enfoca hacia el desarrollo de proyectos que cumplan con los tiempos iniciales establecidos usando una filosofía de desarrollo rápido. Se presentan una serie de errores clásicos que conllevan al fracaso y varios lineamientos que evitan caer en estos.

[Ould 1994]

Ould, Martyn A. *Software Engineer's Reference Book. Capítulo 29. Quality control and Assurance.* Heinemann Ltd., Butterworth, Gran Bretaña, 1994.

En este capítulo del libro el autor define el término calidad (basado en ISO 8402), presenta una clasificación para los requerimientos o atributos de software y profundiza en los términos de aseguramiento y control de calidad. Para este último punto presenta técnicas para inspecciones (Recorridos Estructurados de Ed Yourdon y las inspecciones de Fagan). El capítulo hace referencia en su mayoría a la norma ISO 9000.

[Rakos 1990]

Rakos, John J. *Software Project management for small to medium sized projects*. Prentice-Hall, Inc. EEUU, 1990.

El tema principal de este libro es el método de desarrollo que se debe utilizar para proyectos informáticos pequeños y medianos. La explicación del método se complementa con ejemplos señalando los errores con el fin de evitarlos.

[Robertson 2002]

Robertson, James. *Eureka! Why Analysts Should Invent Requirements*. IEEE Software July/August, 2002, pp. 20-22.

Propone un punto de vista disidente: alentar a los analistas de requerimientos a crear productos más valiosos y competitivos, al inventar requerimientos en lugar de esperar a que los clientes (o usuarios) se los pidan.

[Sawyer 1998]

Sawyer, Pete y Ian Sommerville. *Requirements Engineering: A Good Practice Guide*. John Wiley & Sons Ltd, Inglaterra, 1998.

Ofrece lineamientos o buenas prácticas para mejorar el proceso de ingeniería de requerimientos en las organizaciones. Las sugerencias de este libro son prácticas y se encuentran categorizadas de acuerdo con el nivel de dificultad para aplicarlas.

[Trejos 1997]

Trejos, Ignacio. *Aseguramiento de la Calidad del software 1: Principios y técnicas estáticas*. Instituto Tecnológico de Costa Rica, Centro de Investigaciones en Computación. Cartago, Costa Rica. Marzo, 1997.

En este informe el autor presenta los principios del aseguramiento de calidad aplicados a la construcción de los sistemas de información y profundiza en el empleo de revisiones estáticas para lograr un nivel alto de calidad. Las técnicas descritas detalladamente en este documento son:

1. Vistazos generales
2. Revisiones técnicas
3. Inspecciones
4. Revisiones administrativas
5. Revisiones informales

Este documento se basa principalmente en las normas del IEEE (Institute of Electrical and Electronic Engineers) referentes a la calidad en el software y el libro *Software Inspection Process* de Strauss y Ebenau.

[Trejos 1999]

Trejos, Ignacio; Luna, Antonio. *El modelo de objetos: análisis y diseño*. Club de Investigación Tecnológica, 1999.

Este informe presenta una síntesis de las más importantes ideas del paradigma de orientación a objetos, haciendo énfasis en la formulación de modelos pertinentes al análisis y diseño de sistemas.

[Wiegers 1999]

Wiegers, Karl E. *Software Requirements*. Microsoft Press, EEUU, 1999.

Este libro presenta técnicas para recolectar y administrar los requerimientos a lo largo del ciclo del desarrollo del producto. Se divide en tres partes principales, la primera define aspectos de la ingeniería de requerimientos y de los propios requerimientos. La segunda parte, se relaciona con los procesos para recolectar, documentar y validar los requerimientos. La última parte se enfoca en la administración de los requerimientos a lo largo del ciclo de vida.

[Wilson 1997]

Wilson, Willian M. *Writing Effective Requirements Specifications*. Software Assurance Technology Center, NASA, EEUU, 1997. Tomado de internet http://satc.gsfc.nasa.gov.

Este artículo explica las características que deben tener los documentos de requerimientos y cómo se pueden utilizar métricas para detectar de forma automática las debilidades de estos documentos.