INGENIERÍA DE REQUERIMIENTOS

Autor: Gustavo Rodríguez Diez

La ingeniería de requerimientos es una de las partes más importantes y menos apreciadas de la ingeniería de sistemas y de la ingeniería de software. Como todas las ramas de la ingeniería relativas a la fabricación de software su origen es muy reciente y es hasta hace poco tiempo que es reconocida por los principales autores como una disciplina formal.

El auge que esta disciplina ha tenido en los últimos años se debe a la importante influencia que los requerimientos muestran en las estadísticas de proyectos fracasados o con problemas de tiempo o presupuesto. Lo anterior no sólo ha producido mucha literatura sino que ha obligado a los "metodólogos" a concederle un papel muy importante dentro de sus metodologías al proceso de identificación y captura de requerimientos.

Podemos definir la ingeniería de requerimientos como la ciencia y la disciplina de identificar, documentar, verificar y administrar los requerimientos de un sistema. En este contexto, el mayor objetivo de la ingeniería de requerimientos es definir el propósito de un sistema y capturar el comportamiento externo del mismo.

En la ingeniería de requerimientos distinguimos los "requerimientos del sistema" de los "requerimientos del software". Los primeros se refieren a transformar las necesidades de los clientes y usuarios en requerimientos del sistema, mientras que los segundos se refieren a descomponer el problema en varios problemas pequeños o "subsistemas", en distribuir los requerimientos en dichos subsistemas y en transformar los requerimientos en una especificación más detallada apropiada para los diseñadores y arquitectos de software.

Managing Software Requirements, D. Leffingwell, AW2000

El modelo que se reproduce arriba permite entender mejor la relación entre el sistema y el software. En la parte superior de la pirámide se encuentran las necesidades de los clientes y de los usuarios. Aquí se concentra cualquier área de oportunidad, problema del negocio, problema personal o problema operacional que, si es resuelto por un sistema, produce un beneficio a la compañía, de tal forma que se justifica su construcción o compra. Un ejemplo de una necesidad de un cliente puede ser: "*Necesito disminuir el tiempo de atención de los clientes que hablan por teléfono a mis oficinas*".

Cuando se identifican necesidades es importante hacer una distinción entre clientes y usuarios que nos ayude a identificarlos en el contexto de un proyecto, así como a establecer prioridades y criterios sobre la importancia relativa de cada una de sus necesidades con respecto al aspecto del sistema que estamos analizando. Un cliente es cualquier persona que tiene requerimientos sobre el proyecto, pero que no va a usar el producto de manera directa por ejemplo: el patrocinador, el gerente de mercadotecnia de la empresa para la que trabajamos, la persona que paga por un desarrollo a la medida, el gerente de sistemas de la empresa que va a comprar un desarrollo, etc.

Un usuario es un individuo que va a interactuar directamente con el sistema que se va a construir. Una comunidad de usuarios es un grupo de usuarios que tienen el mismo rol dentro del sistema.

En el área de en medio de la pirámide se encuentra la funcionalidad del sistema. La funcionalidad se define como los servicios que el sistema provee para resolver una o más necesidades de los clientes o usuarios. Es claro que la funcionalidad tiene una relación directa con las necesidades, por lo tanto la funcionalidad debe estar expresada en un lenguaje común y a un nivel lo suficientemente alto como para ser discutidas con los clientes o usuarios y lograr un común acuerdo sobre su significado. Un ejemplo de funcionalidad para la necesidad *"disminuir el tiempo de atención...."* puede ser: *"El sistema debe proveer una lista de los productos que más consume mi cliente"*.

Finalmente, en la base de la pirámide se encuentran los requerimientos de software. Los requerimientos de software pueden ser:

Una capacidad que el sistema necesita para resolver un problema o alcanzar un objetivo.

Una capacidad que el sistema debe poseer para satisfacer un contrato, un estándar, una especificación u otra imposición formalmente documentada.

Los requerimientos se derivan de las funcionalidades, pero son lo suficientemente específicos como para poder derivar código a partir de ellos.

Un ejemplo de requerimientos para el caso anterior:

"La lista de productos debe desplegarse en una tabla que contenga una columna con el identificador del producto y otra con una breve descripción".

"La lista no debe presentar más de 5 productos".

"La lista debe estar organizada por frecuencia, siendo el producto más frecuentemente solicitado el primero de la lista".

Los requerimientos tienen una importancia fundamental en un proyecto de software, las ventajas que una buena especificación de requerimientos proporcionan incluyen (1):

Permite establecer un acuerdo entre los desarrolladores, clientes y usuarios acerca del trabajo que se va a realizar y sobre los criterios de aceptación del producto terminado.

Permite establecer una base para la estimación del proyecto: (tiempo, costo y recursos)

Mejora las características del producto final como: usabilidad, desempeño, facilidad de mantenimiento y otras)

Permite lograr alcanzar los objetivos con la cantidad óptima de recursos.

Sin embargo, el proceso de obtener los requerimientos de un sistema no es un asunto trivial. La influencia de los requerimientos en el fracaso de los proyectos está bien documentada por varios autores (1, 2). Las estadísticas muestran que los requerimientos o alguna actividad relacionada con los mismos es identificada como una de las causas principales de fracaso en proyecto de software.

Existen diversas situaciones que pueden provocar una deficiente identificación y captura de requerimientos y provocar la catástrofe de un proyecto. La mayoría de las veces estas deficiencias están relacionadas con la falta de importancia que en muchas organizaciones se le da al proceso de especificación de requerimientos. Esta falta de importancia puede traducirse en situaciones como las siguientes (1):

La presión que muchos clientes ejercen para reducir o no pagar el tiempo de captura de requerimientos ya que consideran que el verdadero valor se obtiene a través de las actividades de programación de código.

La falta de cooperación del cliente para verificar que los requerimientos capturados son correctos y la falta de comprensión de la importancia de las especificaciones.

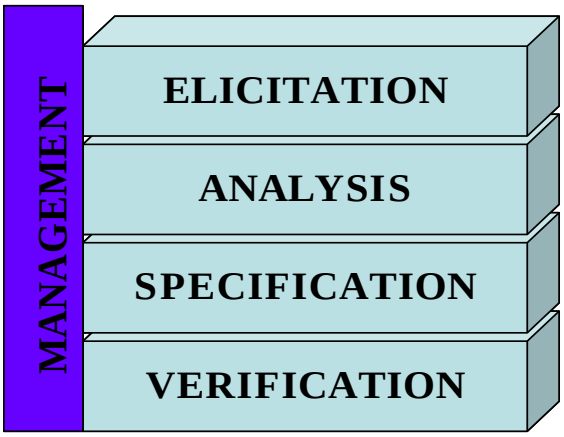
La subestimación de la importancia que los requerimientos tienen para todo el proyecto.

La falta de entrenamiento formal de los analistas en técnicas para la identificación y captura de requerimientos.

La práctica errada de muchos gerentes de poner a personal con poca experiencia y conocimiento para desarrollar software en posiciones clave o de gran responsabilidad donde provocan quiebres al proceso de desarrollo de grandes proyectos.

Es bien conocido que los proyectos que son intensivos en software son más complejos en su naturaleza que los proyectos que no lo son. Adicionalmente, los problemas para generar una buena especificación aumentan cuando el tamaño o la complejidad del proyecto aumentan. En este escenario la elección de técnicas o herramientas adecuadas para especificar requerimientos y la escritura de una especificación de software adecuada y completa se vuelven muy complejas.

Ante todos estos retos y dada la importancia que tienen para un proyecto producir una buena especificación de software, surge la necesidad de establecer una disciplina que ayude a enfrentar con seriedad la problemática del desarrollo de requerimientos. La ingeniería de requerimientos debe enfocarse a resolver las preguntas de: ¿Cómo deben desarrollarse los requerimientos de un sistema?, ¿Cómo podemos evaluar que los requerimientos de un sistema están completos?, ¿Qué estándares, herramientas y métodos pueden ayudar con el problema de los requerimientos?. La ingeniería de requerimientos se enfoca, no sólo en el análisis y especificación de requerimientos de un sistema, sino también en la investigación abstracta de las mejores formas para desarrollar los requerimientos.



Varias taxonomías han sido propuestas para la ingeniería de requerimientos, sin embargo la mayoría de los trabajos al respecto están de acuerdo en el modelo que propone Dorfman(1)(3) o en alguna variación de este mismo modelo:

Elicitación: Es el proceso mediante el cual el cliente y el analista de un sistema de software descubren, revisan, articulan y entienden las necesidades de clientes y usuario y las restricciones del software y de la actividad de desarrollo.

Análisis: Es el proceso de analizar las necesidades de los clientes y de los usuarios para llegar a la definición de los requerimientos del software.

Especificación: Es el desarrollo de un documento que, de manera clara y precisa, registre cada uno de los requerimientos del software.

Verificación: Es el proceso de asegurar que la especificación de requerimientos cumple con las necesidades de los clientes y usuarios, cumple con los estándares definidos en la organización y es una base adecuada para establecer la arquitectura preliminar del proyecto.

Administración: Es la planeación y control de las actividades que ocurren durante el proceso de desarrollo de los requerimientos.

Con excepción de la administración, las partes se mencionan en el mismo orden lógico en el que se aplican en un proyecto. Un proceso de ingeniería de software debe incluir en la descripción del flujo de trabajo de requerimientos, actividades en cada una de las áreas señaladas para logra una aproximación métodica y progresiva a la especificación del proyecto.

Las actividades de administración de los requerimientos son las mismas que se realizan en la administración general de proyectos de software, la diferencia es que los planes que se formulan y las actividades de control son específicos para el desarrollo de requerimientos. La administración de requerimientos es una actividad de soporte que ocurre a lo largo de todo el

proceso de desarrollo de requerimientos y por lo tanto se coloca de manera vertical en el modelo.

Existen diferentes técnicas que pueden aplicarse para lograr los objetivos particulares de cada una de las partes de la ingeniería de requerimientos. A continuación, se mencionan algunas de las más usadas.

Elicitación

Como ya hemos visto, la elicitación consiste en identificar las necesidades de los usuarios y las restricciones del sistema. De acuerdo al modelo de la pirámide de requerimientos podemos considerar que elicitación del proyecto se queda en el nivel de funcionalidad y deja para la parte de análisis el determinar los requerimientos del sistema.

La elicitación es un proceso complejo, ya que se enfrenta con algunos síndromes que han sido definidos como las "barreras de la elicitación" (1). Estos síndromes son situaciones implícitas al proceso de desarrollo de software que complican la administración y agregan un factor de incertidumbre al proyecto.

El síndrome de "si, pero..."

Este síndrome es uno de los problemas más frustrantes y persistentes en un proyecto de software. El síndrome consiste en que por alguna razón cuando presentamos el sistema al cliente por primera vez y en caso de que el sistema sea lo que el cliente espera, ocurren dos reacciones opuestas y paralelas: Por un lado el cliente está contento y nos felicita por nuestro esfuerzo e incluso menciona situaciones en las que el sistema será de mucho valor. Por otra parte, el cliente dice: Pero ¿que pasa en esta situación?, o ¿no sería mejor que esta operación se hiciera de otra manera?, o aún mejor: ¿No podríamos agregar esta funcionalidad?.

La causa de este síndrome es parte de la naturaleza mísma del software como un proceso intelectual intangible. La reacción del usuario es parte de la naturaleza humana y ocurre en otras situaciones del día a día, el motivo es que después de muchos meses de desarrollo ahora es capaz de entender que es lo que el desarrollador quería decir cuando explicaba una u otra funcionalidad del sistema. De la misma manera que una imagen dice más que mil palabras, el producto que presentamos al usuario es mucho más explícito que su especificación.

La manera en la que podemos reducir significativamente el impacto de este síndrome es aplicar técnicas que permitan que los "peros" de clientes y usuarios se produzcan cuanto antes en el proceso de desarrollo.

El síndrome de "las ruinas enterradas".

De alguna manera, la búsqueda de requerimientos es como la búsqueda de ruinas arqueológicas: "entre más descubres, sabes que todavía queda mucho más enterrado". Cuando se identifican los requerimientos en un proyecto de software no es posible llegar a un punto en el cual uno sienta que ha identificado todo lo que había por encontrar.

Este síndrome provoca que los analistas enfrenten continuamente la pregunta de ¿serán suficientes los requerimientos identificados para comenzar el desarrollo?. Esto puede llevar a otro síndrome conocido en la administración de proyectos como "parálisis por análisis", que consiste en que el proyecto se estanca indefinidamente en un proceso de identificación de requerimientos que paraliza al proyecto hasta provocar su cancelación.

El Proceso de Elicitación

El proceso de elicitación puede describirse en cuatro pasos(4):

Identificar el problema.

Definir las fronteras de la solución.

Identificar las restricciones impuestas a la solución.

Entender las necesidades de los clientes y usuarios.

Identificar el problema.

El proceso de elicitación inicia con la identificación del problema que se va a resolver. Es importante que la definición del problema sea consensada por el cliente y todos los

involucrados en el proceso de desarrollo. La manera más sencilla de lograr esto es escribir el problema y modificarlo hasta que se logre un consenso.

Un formato que puede ayudarnos a identificar el problema es el siguiente (1):

El problema de: *(Descripción del problema)*

Afecta a: *(Identificar a los usuarios y clientes afectados por el problema)*

En qué: *(Describir el impacto del problema en los afectados y en las actividades del negocio)*

Una solución exitosa es: *(Indicar la solución propuesta y listar los principales beneficios)*

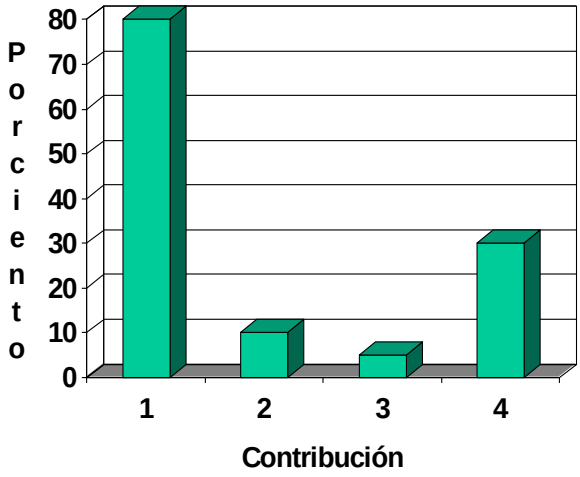
Es común que en el proceso de identificación del problema, la gente pregunte si el problema debe ser sólo uno. La respuesta es sí, cada proyecto debe de tener un sólo enunciado del problema, por tanto el enunciado debe ser el de más alto orden encontrado.

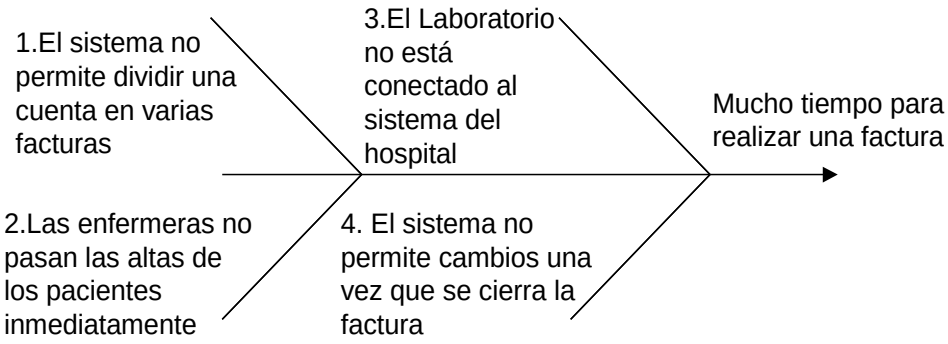
Un error frecuente es que los analistas confunden el verdadero problema con los efectos finales o los síntomas que el problema produce, en estos casos es útil preguntar ¿por qué sucede esto?, hasta llegar a lo que se puede considerar la raíz del problema o el problema detrás del problema.

Una técnica útil para descubrir el problema detrás del problema son los diagramas de cola de pescado y los diagramas de pareto.

Por ejemplo:

> En un hospital se tiene problemas con el tiempo para generar la factura de los pacientes dados de alta. El departamento de quejas recibe muchas protestas de clientes sobre el tiempo en el que se tardan en entregarles su factura y el departamento de mercadotecnia ha detectado que se registra una fuerte pérdida de clientes por esta causa.
>
> El director del hospital llegó a la conclusión de que es necesario comprar un nuevo sistema de facturación para evitar seguir perdiendo clientes. Una investigación en el mercado de ERP para hospitales da como resultado que el presupuesto y el tiempo de implementación de esta solución, la dejan fuera de las necesidades actuales del hospital.
>
> El director entonces, invita a una empresa de consultoría y desarrollo para que lo ayude a encontrar una solución. La empresa comienza el proceso de elicitación y descubre que las causas del problema de facturación son las siguientes:
>
> > El sistema actual no permite dividir una cuenta en varias facturas. Cuando un paciente quiere reclamar los cargos de su cuenta que cubre su seguro a su compañía aseguradora, el cajero tiene que cancelar la cuenta del paciente y posteriormente crear dos cuentas separadas para facturar.
> >
> > En ocasiones los médicos avisan del alta al paciente antes que a la enfermera encargada, por lo que la enfermera no envía el alta del paciente a la caja y la factura se demora.
> >
> > El área de Laboratorio no está conectada al sistema del hospital por lo que los cargos a la cuenta del paciente se hacen a través del área de farmacia. Debido que estos cargos no se hacen en línea el cajero tiene que hablar a farmacia para saber si el paciente no tiene cargos pendientes.
> >
> > El sistema no permite cambios una vez que se cierra la factura. En ocasiones algún área del hospital informa de algún cargo pendiente después de que el cajero cerró la factura, o el cliente nota que su nombre o su dirección están incorrectas. Esto provoca que la factura tenga que ser cancelada y creada nuevamente.





Con esta información, la empresa consultora genera estadísticas de una muestra representativa de facturas y descubre lo siguiente: Ya que el 80% de los pacientes entran por medio de una compañía aseguradora, el principal problema de facturación del hospital es la imposibilidad de dividir una cuenta en varias facturas. En la estadística también aparece el segundo problema en importancia es la imposibilidad de hacer cambios a una factura cerrada. En este punto, la empresa consultora, decide reformular el problema de esta manera:

El problema de: Generar cambios en las cuentas y en las facturas de los clientes.

Afecta a: Los clientes en general, la gerencia del hospital.

En qué: Los clientes tienen que esperar hasta una hora para recibir su factura, lo que provoca que no recomienden al hospital y que elijan otro hospital para atenderse.

Una solución exitosa es: Modificar el sistema actual para que permita:

Generar varias facturas a partir de la misma cuenta.

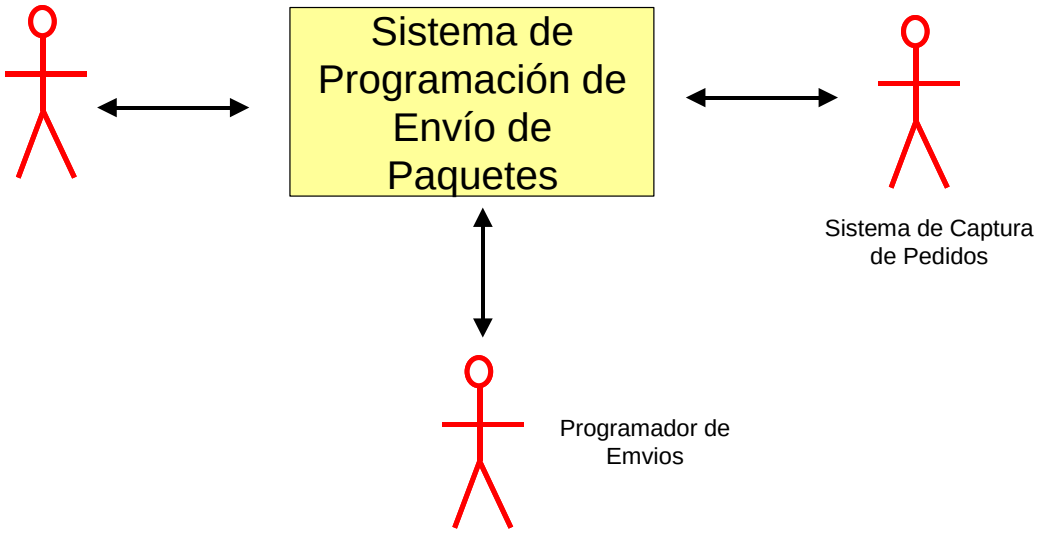
Realizar cambios a una factura cerrada.

Definir las fronteras de la solución.

Una vez que se ha alcanzado un acuerdo sobre el problema que se va a construir, es necesario definir un sistema que pueda ser construido para resolver el problema. Para hacer esta tarea debemos comenzar con establecer las fronteras dentro de las cuales se creará el sistema.

Una manera de acotar un sistema es ponerlo en el contexto de los elementos externos que van a interactuar con él. Los elementos externos pueden ser: humanos, elementos de hardware u otros sistemas. A estos elementos se les conoce dentro de UML como Actores.

En el inicio de un proyecto es conveniente establecer qué partes de la solución forman parte del proyecto de construcción del sistema, y qué partes van a interactuar con el sistema como elementos externos o actores. Esto facilita que los participantes mantengan el enfoque del proyecto y trabajen sólo dentro del alcance del mismo; adicionalmente permite administrar de manera correcta las expectativas del cliente.



Una herramienta que nos permite documentar las fronteras del proyecto es el diagrama de contexto. Un diagrama de contexto emplea un rectángulo para definir el proceso y utiliza figuras de palitos para representar a los actores (lo cual es un estándar de UML).

Identificar las Restricciones Impuestas a la Solución

Una restricción es una reducción al grado de libertad que tenemos para generar una solución. Existen diferentes tipos de restricciones que pueden presentarse en un proyecto.

Cada restricción puede potencialmente restringir nuestra habilidad para desarrollar una solución tal y como la visualizamos. Por lo tanto cada restricción debe considerarse dentro del proceso de planeación para analizar su impacto y determinar qué estrategia se empleará para cumplirla.

Como una ayuda al proceso de identificar restricciones se presenta la siguiente lista (4):

Económicas

¿Qué restricciones financieras o de presupuesto son aplicables?

¿Existe alguna consideración de precios de productos?

¿Existe alguna restricción de licencias?

¿Una falla puede interrumpir o dañar las operaciones diarias críticas del negocio?

¿Puede este proyecto incurrir o causar perdidas financieras significantes?

¿Es este un esfuerzo grande (> 6 meses o $100,000 Dls.)?

Políticas

¿Existen cuestiones políticas internas o externas que puedan afectar la solución?

¿Existen problemas o cuestiones interdepartamentales que puedan afectar la solución?

¿Fallar en el proyecto puede dañar la reputación de la empresa?

¿Este problema no ha podido ser resuelto en el pasado?

¿Existe algún participante que se oponga o tenga muchas dudas del proyecto?

Existirán más de 5 personas en el equipo del proyecto o en el "Steering Comitee"?

Técnicas

¿Existe alguna restricción en la elección de la tecnología?

¿Existe alguna restricción para trabajar con las plataformas o técnicas existentes?

¿Está restringido el uso de alguna nueva tecnología?

¿Es necesario usar algún paquete de software adquirido por el cliente?

¿El producto depende de tecnología experimental?

Si lo anterior ocurre, ¿estará involucrao más de un proveedor o componente crítico?

¿Existe un alto nivel de complejidad técnica involucrado?

Sistemas

¿La solución se construirá sobre un sistema existente?

¿Se debe mantener la compatibilidad con alguna solución existente?

¿Qué sistemas operativos y ambientes deben ser soportados?

Ambientales

¿Existen restricciones regulatorias?

¿Existen requerimientos de seguridad?

¿Qué otros estándares debe respetar el proyecto?

¿Existen restricciones legales o ambientales?

¿Está involucrada más de una empresa?

¿Más de una empresa será impactada por el producto?

Calendario y Recursos

¿El calendario del proyecto está definido?

¿Se está restringido a los recursos actuales?

¿Pueden usarse externos?

¿Pueden expandirse los recursos temporal o permanentemente?

¿Este proyecto está en busca de un campeón?

¿Este proyecto está en busca de un líder o administrador?

¿El proyecto entrará en un calendario acelerado o comprometido?

¿Lo anterior representa una dificultad o problema a largo plazo?

¿Es el primer esfuerzo de esta compañía en proyectos de este tipo?

¿Algún contratista externo liderará o producirá algún entregable clave?

¿Es necesario establecer un plan o asignar responsabilidades?

¿El equipo de trabajo carece de alguna habilidad necesaria?

¿Los recursos del proyecto están administrados /controlados por más de una persona?

¿Los miembros clave del equipo se encuentran en departamentos o edificios separados?

¿Hay dudas acerca del compromiso o disponibilidad de algún recurso clave?

Una vez identificadas algunas de las restricciones se volverán requerimientos del sistema que vamos a construir, otras restricciones afectarán recursos, planes de implementación y planes del proyecto. Es responsabilidad del analista entender las fuentes potenciales para cada restricción y determinar el impacto de cada una en el espacio de la solución del problema.

Entender las necesidades de clientes y usuarios

La solución de un problema complejo, normalmente implica satisfacer las necesidades de diversos grupos de interés. Estos grupos pueden ser divididos en clientes y usuarios: Un *cliente* es una persona que tiene influencia sobre los requerimientos del sistema aunque no vaya a ser un usuario del mismo, mientras que los *usuarios* son las personas o grupos de personas que van a interactuar directamente con el sistema.

Entender las necesidades de clientes y usuarios es una parte fundamental de la elicitación de un proyecto. Para entender las necesidades de estas personas hay que empezar por identificarlos. Algunas preguntas que podemos hacer para ayudarnos en esta labor son:

¿Cuáles serán los diferentes roles organizacionales que usaran el sistema?

¿Quién va a pagar por el sistema?

¿Qué otra persona se verá afectada por las salidas que el sistema produce?

¿Quién es responsable de evaluar y aceptar el sistema cuando se libere?

¿Quién será responsable de darle mantenimiento al sistema?

Una vez que hemos identificado a los clientes y usuarios podemos aplicar varias técnicas para identificar cuáles son sus necesidades. La siguiente sección describe esas técnicas y algunas herramientas que ayudan a entender cuáles son las necesidades de los usuarios respecto al sistema que se va a construir.

Técnicas de Elicitación

Una cuestión básica en Ingeniería de Requerimientos es cómo determinar qué es lo que el cliente y los usuarios verdaderamente necesitan. Este proceso ha demostrado en la práctica que es muy complejo y que puede fácilmente provocar el fracaso de un proyecto.

Existen varias técnicas que pueden utilizarse para la elicitación de un sistema, sin embargo, dada la naturaleza social del proceso, el uso de las mismas debe considerar no sólo los aspectos técnicos sino los aspectos sociales, políticos y culturales del proyecto. Normalmente, el aspecto social del proceso no es fácil de describir en primera instancia, requiere una inmersión profunda en el ámbito de los individuos, no es suficiente la simple recolección de información, como por ejemplo, reunir estadísticas de la ocurrencia de ciertas categorias predeterminadas.

Introspección

La introspección es el medio más obvio y comúnmente usado para entender los requerimientos de un sistema. Consiste en estudiar el ambiente del usuario para posteriormente tratar de imaginar qué es lo que me gustaría que hiciera el sistema si yo estuviera a cargo de hacer el trabajo con su ayuda.

Esta técnica es útil pero hay que considerar que la experiencia de un analista de sistemas es muy diferente que la experiencia de los usuarios potenciales del software y por lo tanto es difícil que las conclusiones del analista reflejen la experiencia de las personas que hacen la tarea día con día.

Esto se hace más evidente en el diseño de la interfaz gráfica del usuario, lo que para un analista es el comportamiento común de un software, puede resultar confuso o frustrante para un usuario. Tradicionalmente, debido a las limitaciones gráficas que anteriormente tenían las computadoras, se creaba una brecha entre el diseño de la aplicación y las necesidades de usabilidad del usuario, sin embargo debido a la evolución de los ambientes gráficos el paradigma está cambiando: *La obligación del usuario no es aprender a dominar una tecnología, es la tecnología la que debe ayudar al usuario a hacer su trabajo*. La explosión del Internet ha

creado nuevas especialidades como "user experience" en las cuales la psicología y la sociología son las principales herramientas científicas.

La recomendación, cuando se usa la introspección, es apoyarla con la información previa que generan otras técnicas como por ejemplo *la entrevista abierta*. En cualquier caso es recomendable validar cualquier duda con un "experto de dominio". Los expertos de dominio son personas con mucha experiencia en un área particular del negocio que es de interés para el sistema. Por ejemplo, en el caso de una línea de producción, el operador de alguna máquina de la línea puede ser un experto de dominio que aporte valiosa información para la automatización del proceso.

Entrevista Abierta

Una de las técnicas más importantes y más directas para obtener requerimientos es la entrevista abierta. El proceso de realizar una entrevista puede parecer trivial en muchos casos, sin embargo, debido al factor social que ya hemos mencionado y al "síndrome del usuario y el desarrollador", las entrevistas pueden presentar complicaciones.

Una de las principales consideraciones a tomar en cuenta en una entrevista es lograr que la predisposición del entrevistador no influya en el resultado de la narrativa capturada. Como proveedores de soluciones, estamos acostumbrados a identificar soluciones al mismo tiempo que escuchamos problemas, cuando hacemos esto, tendemos a influenciar las respuestas del entrevistado, provocando una mala comprensión del problema o una reducción en el grado de libertad de la solución.

Una entrevista debe considerar preguntas libres de contexto (Gausse and Weinberg, 1989), es decir, preguntas que no influencien las respuestas de los entrevistados hacia las respuestas que queremos oír. Por ejemplo:

¿Quiénes son los usuarios del sistema?

¿Qué problemas tienen actualmente?

¿Cuáles son sus necesidades respecto a la solución?

Este tipo de preguntas fuerzan al entrevistador a escuchar antes de identificar una solución potencial, así mismo permiten al entrevistado explayarse en la descripción de la problemática que enfrenta. Cuando el analista escucha lo que el usuario tiene que decir respecto a sus necesidades, se produce un mejor entendimiento del problema y por tanto una solución adecuada.

Las preguntas libres de contexto son una técnica similar a la que los profesionales de ventas utilizan como parte de la estrategia conocida como "solution selling". En esta estrategia el vendedor pregunta para obtener información sobre la problemática del cliente antes de ofrecer un producto o un servicio. Cuando el vendedor ha obtenido una visión clara de la problemática del cliente, entonces propone algo que tiene sentido para el cliente y entonces la venta se produce más fácilmente.

Pasos para una entrevista abierta

Una entrevista abierta requiere prepararse con anterioridad. Una recomendación básica es preparar una guía con preguntas apropiadas que permitan mantener una conversación fluida a la vez que ayuden a asegurar que no se olvida tocar ningún tema importante durante la entrevista. La guía no debe seguirse al pié de la letra, simplemente es un apoyo para usarse durante la entrevista. A continuación se listan algunos pasos para preparar una entrevista.

Investigar el background del usuario y de la empresa antes de la entrevista

Revisar las preguntas antes de la entrevista

Consultar la lista de preguntas durante la entrevista para asegurarse de hacer las preguntas correctas

Al final de la entrevista, identificar los dos o tres problemas principales y repetir lo que se entendió para asegurar la comprensión

Algunos consejos que pueden ayudarnos durante una entrevista:

No hagas preguntas donde se suponga de antemano que la gente puede describir actividades complejas. En general la gente puede hacer muchas cosas que no puede describir. Cuando necesites entender una actividad compleja separa la pregunta en varias partes o utiliza otra técnica como la investigación de campo.

Haz preguntas abiertas que le permitan al usuario extenderse en sus respuestas y a ti comprender mejor su problemática.

No hagas preguntas que influencíen la respuesta del entrevistado: ¿Usted necesita una pantalla para realizar esta tarea, verdad?

No hagas preguntas para controlar la conversación: ¿Podemos regresar a la pregunta anterior?

No hagas preguntas que se respondan así mismas: ¿50 elementos en esta lista son suficientes, verdad?

Evita preguntas que empiecen con ¿Por qué?. Habitualmente estas preguntas ponen al entrevistado a la defensiva porque aparentan cuestionar la manera en que hace su trabajo.

No esperes respuestas sencillas. Cuando las obtengas sigue preguntando para descubrir más "ruinas enterradas".

No apures al entrevistado para que responda. Si haces esto probablemente no querrá responder tu próxima pregunta.

Por último lo más importante: Escucha con atención.

Cuestionarios

Los cuestionarios son otra técnica de elicitación. Consisten en una serie de preguntas que el entrevistador hace de manera secuencial o que se entregan al entrevistado para que él mismo conteste.

Los cuestionarios tienen la deficiencia de que no utilizan los elementos de interacción de la entrevista, por lo tanto se corre el riesgo de que, dado que la persona a la que se entrevista tiene diferente historia y por lo tanto diferentes conocimientos y categorías para clasificar los conceptos, algunas preguntas se malinterpreten o resulten irrelevantes y fuera de contexto.

Grupos de Desarrollo de Aplicaciones

Consiste en involucrar a los usuarios en el proceso de desarrollo mediante talleres o workshops en los cuales se identifican los requerimientos. En los talleres pueden utilizarse diferentes técnicas para ir descubriendo requerimientos como "Casos de Uso", "Story Boards", "Modelos" o "Prototipos".

Los grupos de desarrollo tienen el inconveniente de que no son comunidades naturales, por lo tanto es difícil que los participantes compartan categorías. Otro riesgo en estos talleres es que, dadas las jerarquías que existen dentro de una empresa, alguno de los participantes puede no sentirse libre para expresar su opinión o puede sentirse obligado a dar una opinión sobre un punto que desconoce o en el que él no es experto.

Algunas recomendaciones para conducir un grupo de desarrollo son:

Da a todos la oportunidad de hablar. Esto es esencial para que el workshop se considere imparcial.

Procura que la sesión se mantenga andando. Existe una tendencia natural a que el workshop se convierta en una "sesión de quejas". Identifica los problemas/requerimientos pero no profundices. Una vez que se ha identificado un problema/requerimiento avanza al siguiente.

Permanece alerta para recoger información y hallazgos.

Al final, resume la sesión y trabaja en generar conclusiones

Tormenta de ideas

Cuando se está conduciendo una sesión de requerimientos es necesario en ocasiones usar alguna herramienta que permita identificar y clasificar necesidades. Una herramienta muy útil para lograr esto es la "tormenta de ideas". La Tormenta de Ideas consiste en listar todas las

ideas sobre un tema que se le ocurren a un auditorio determinado y posteriormente filtrarlas, desarrollarlas y priorizarlas.

Una sesión de este tipo comienza con la aclaración del objetivo. El objetivo es muy importante porque permite mantener en foco la sesión, sin embargo no debe de ser limitante para la creatividad de las ideas expresadas, es más las ideas más descabelladas pueden resultar en soluciones innovadoras. Los participantes deben de aportar ideas sin interrupción y tantas como sea posible, para lograr esto, el moderador debe crear un ambiente en el que la creatividad y la apertura de mente tengan un lugar prioritario, por ejemplo evitando críticas o debates durante la aportación de ideas.

Cuando las ideas comienzan a repetirse y los participantes empiezan a demorarse mucho tiempo entre idea e idea, es momento de suspender la sesión y pasar a filtrar, combinar, ordenar y priorizar las ideas. Al hacer esto, es necesaria la participación del grupo mediante debates y consensos. El moderador debe de evitar que la discusión se vuelva personal o que los debates se prolonguen demasiado, esto puede lograrse usando rondas de votación con prioridades, en las cuales a los miembros se les da una cantidad de puntos que deben distribuir entre las diferentes opciones, nunca asignado más del 50% de sus puntos a una opción. Al final se suman los puntos y la opción con más puntos resulta ganadora.

Storyboards

El Storyboard es una técnica tradicionalmente usada en el cine para describir una secuencia o una escena. Consiste en ilustrar o animar cómo el sistema encaja en la organización para analizar el comportamiento del mismo.

Los storyboard pueden ser tan simples como un par de trazos hechos a lápiz o tan complicados como el diseño gráfico de una página de Web. Sin embargo es recomendable mantenerlos fáciles de hacer y de cambiar, ya que en el fondo debe ser una herramienta barata que ayude a descubrir requerimientos y necesidades por lo tanto su naturaleza será cambiar. Cuando se tienen más claros los requerimientos o cuando se requiere un producto para vender una necesidad a un cliente es más recomendable usar los prototipos y los demos.

Análisis

El análisis de los requerimientos consiste en estudiar las necesidades de los clientes para plantearlas en términos de un sistema de software. Para lograr esto, existen diferentes técnicas como diagramas de contexto, diagramas de flujo o diagramas de estado. UML utiliza la técnica de Casos de Uso para analizar las necesidades de los usuarios y estructurarlas a manera de servicios que el sistema debe proveer.

No debe confundirse el análisis en el contexto de requerimientos y el análisis en el contexto de actividades de análisis y diseño del sistema. En el paradigma de objetos las actividades de análisis y diseño no son siempre claramente diferenciables porque un diagrama de clases puede implicar el qué del sistema y el cómo. Por lo tanto, en el paradigma de objetos, una clasificación para las actividades del ciclo de vida de desarrollo que suele usarse es requerimientos, análisis y diseño y codificación.

Especificación

La especificación consiste en la documentación de las actividades durante el proceso de Requerimientos. De acuerdo a la IEEE(1) existe una serie de atributos que debe tener una especificación de software:

Claridad/Carencia de ambigüedad: Cada uno de los requerimientos tiene una sola interpretación posible.

Completa: Todo lo que el software debe hacer está incluido en las especificaciones. Las respuestas a cualquier tipo de datos de entrada en cualquier situación posible están incluidas en las especificaciones.

Correcta: Cada uno de los requerimientos representa algo que el sistema requiere.

Entendible: Cualquier tipo de lector puede, fácilmente, comprender el significado de todos los requerimientos con una explicación mínima

Verificable: Existe alguna técnica finita y costeable que puede ser usada para verificar que cada uno de los requerimientos especificados está incluido en el sistema terminado.

Consistente: No existen conflictos entre requerimientos.

Factible: Existe por lo menos un diseño y una implementación que satisfacen todos los requerimientos especificados.

Rastreable: Está escrita de manera que facilita la referencia de cada requerimiento individual. El origen de cada requerimiento está claro.

Concisa: Es lo más corta posible, sin afectar otras cualidades de la especificación.

Diseño Independiente: Existe más de un diseño e implementación que satisface correctamente todos los requerimientos del sistema.

Modificable: Tiene una estructura y estilo que permiten hacer cambios de una manera fácil, completa y consistente.

No redundante: Los requerimientos no están repetidos

Precisa: Se usan cantidades numéricas cuando es posible con el apropiado nivel de precisión.

Los atributos son una referencia contra la cual evaluar un documento de requerimientos y no implican una estricta e inflexible comparación. Verificar una especificación de una manera demasiado rígida puede llevar a un proceso muy largo y costoso que, en muchas ocasiones, no es justificable por el tamaño y las características del proyecto.

Además puede suceder que algunos de los atributos se contrapongan entre sí. Por ejemplo, una especificación puede ser factible en el sentido de que hay una única solución identificada pero no puede ser de diseño independiente porque sólo existe una solución identificada. Una especificación nunca llegará a ser perfecta, hay que encontrar un balance adecuado entre la calidad de la especificación y las necesidades del proyecto.

Verificación

Existen varias técnicas para validar una especificación, algunas de las más importantes son:

Verificación a través de rastreo: Implica la revisión de la documentación por parte de un especialista. Algunos autores sugieren que el especialista debe invertir en promedio 30 minutos en revisar una página de documentación.

Pruebas de Validación: Son pruebas que se hacen el software para comprobar que este cumpla con algún requerimiento especificado en la documentación.

Pruebas de Aceptación: Son pruebas que realiza el cliente antes de declarar su satisfacción sobre el producto.

BIBLIOGRAFÍA

Richard H. Thayer, Merlin Dorfman. Software Requirements Engineering, IEEE 1997.

The Standish Group, Chaos Report, http://standishgroup.com/visitor/chaos.htm.

Alan M. Davis. Private comunications, 1996

Dean Leffingwell, Don Widring. Managing Software Requirements (A Unifiend Approach). Addison Wesley 2000.