

11.2. TÉCNICAS DE COMUNICACIÓN

El análisis de requisitos del software siempre empieza con la comunicación entre dos o más partes. Un *cliente* tiene un problema que pretende sea resuelto con una solución basada en computadora.

Un desarrollador responde a la solicitud de ayuda del cliente. La comunicación ha empezado. Pero como ya hemos señalado, el camino de la comunicación al entendimiento está a menudo lleno de baches.

11.2.1. Inicio del proceso

La técnica de análisis más usada para cubrir el hueco de comunicación entre el cliente y el desarrollador y para empezar el proceso de comunicación es llevar a cabo una reunión o entrevista preliminar. La primera reunión entre el ingeniero del software (el analista) y el cliente puede compararse con la primera cita entre dos adolescentes. Nadie sabe qué decir o preguntar; los dos están preocupados de que lo que digan sea malentendido; ambos piensan qué pasará (los dos pueden tener expectativas radicalmente diferentes); los dos están deseando que aquello termine, pero, al mismo tiempo, ambos desean que la cita sea un éxito.

Sin embargo, hay que empezar la comunicación. Gause y Weinberg [GAU89] sugieren que el analista empiece preguntando *cuestiones de contexto libre*. Es decir, un conjunto de preguntas que llevarán a un entendimiento básico del problema, qué solución busca, la naturaleza de la solución que se desea y la efectividad del primer encuentro. El primer conjunto de cuestiones de contexto libre se enfoca sobre el cliente, los objetivos generales y los beneficios esperados. Por ejemplo, el analista podría preguntar:

- ¿Quién está detrás de la solicitud de este trabajo?
- ¿Quién utilizará la solución?
- ¿Cuál será el beneficio económico del éxito de una solución?
- ¿Hay alguna otra alternativa para la solución que necesita?

El siguiente conjunto de preguntas permite al analista obtener un mejor entendimiento del problema y al cliente comentar sus opiniones sobre la solución:

- ¿Cómo caracterizaría una «buena» salida (resultado) generada para una buena solución?
- ¿A qué tipo de problema(s) va dirigida esta solución?
- ¿Puede mostrarme (o describirme) el entorno en que se utilizará la solución?

- ¿Hay aspectos o restricciones especiales del rendimiento que afecten a la manera de enfocar la solución?

El último conjunto de preguntas se concentra en la eficacia de la reunión. Gause y Weinberg [GAU89] las denominan «meta-preguntas» y proponen la siguiente lista (abreviada):

- ¿Es usted la persona adecuada para responder a estas preguntas? ¿Sus respuestas son «oficiales»?
- ¿Son adecuadas mis preguntas para el problema que tiene usted?
- ¿Estoy preguntando demasiado?
- ¿Hay alguien más que pueda proporcionar información adicional?
- ¿Hay algo más que debería preguntarle?

Estas preguntas (y otras) ayudarán a «romper el hielo» e iniciar la comunicación tan esencial para el éxito del análisis. Pero el formato de reunión tipo «pregunta y respuesta» no es un enfoque que haya tenido mucho éxito. De hecho, esta sesión de preguntas y respuestas debería emplearse solamente en el primer encuentro y sustituirse después por una modalidad que combine elementos de resolución del problema, negociación y especificación. En la siguiente sección se presenta un enfoque a reuniones de este tipo.

11.2.2. Técnicas para facilitar las especificaciones de una aplicación

Los clientes y los ingenieros del software a menudo tienen una mentalidad inconsciente de «nosotros y ellos». En vez de trabajar como un equipo para identificar y refinar los requisitos, cada uno define por derecho su propio «territorio» y se comunica a través de una serie de memorandos, papeles de posiciones formales, documentos y sesiones de preguntas y respuestas. La historia ha demostrado que este método no funciona muy bien. Abundan los malentendidos, se omite información importante y nunca se establece una buena relación de trabajo.

Con estos problemas en mente es por lo que algunos investigadores independientes han desarrollado un enfoque orientado al equipo para la reunión de requisitos que se aplica durante las primeras fases del análisis y especificación. Denominadas *Técnicas para facilitar las especificaciones de la aplicación* (TFEA), este enfoque es partidario de la creación de un equipo conjunto de clientes y desarrolladores que trabajan juntos para identificar el problema, proponer soluciones, negociar diferentes enfoques y especificar un conjunto preliminar de requisitos de la solución [ZAH90]. Hoy en día, las TFEA son empleadas de forma general por los sistemas de

información, pero la técnica ofrece un potencial de mejora en aplicaciones de todo tipo.

Se han propuesto muchos enfoques diferentes de TFEA³. Cada uno utiliza un escenario ligeramente diferente, pero todos aplican alguna variación en las siguientes directrices básicas:

- La reunión se celebra en un lugar neutral y acuden tanto los clientes como los desarrolladores.
- Se establecen normas de preparación y de participación.
- Se sugiere una agenda lo suficientemente formal como para cubrir todos los puntos importantes pero lo suficientemente informal como para animar el libre flujo de ideas.
- Un «coordinador» (que puede ser un cliente, un desarrollador o un tercero) que controle la reunión.
- Se usa un «mecanismo de definición» (que puede ser hojas de trabajo, gráficos, carteles o pizarras).
- El objetivo es identificar el problema, proponer elementos de solución, negociar diferentes enfoques y especificar un conjunto preliminar de requisitos de la solución en una atmósfera que permita alcanzar el objetivo.

Para comprender mejor el flujo de acontecimientos tal y como ocurren en una reunión TFEA, presentamos un pequeño escenario que esboza la secuencia de acontecimientos que llevan a la reunión, los que ocurren en la reunión y los que la siguen.

En las reuniones iniciales entre el desarrollador y el cliente (Sección 11.2.1) se dan preguntas y respuestas básicas que ayudan a establecer el ámbito del problema y la percepción global de una solución. Fuera de estas reuniones iniciales, el cliente y el desarrollador escriben una «solicitud de producto» de una o dos páginas. Se selecciona lugar, fecha y hora de reunión TFEA y se elige un *coordinador*. Se invita a participar a representantes de ambas organizaciones; la del cliente y la de desarrollo. Se distribuye la solicitud de producto a los asistentes antes de la fecha de reunión.

Mientras se revisa la solicitud días antes de la reunión, se pide a todos los asistentes que hagan una lista de objetos que formen parte del entorno del sistema, otros objetos que debe producir el sistema y objetos que usa el sistema para desarrollar sus funciones. Además, se pide a todos los asistentes que hagan otra lista de *servicios* (procesos o funciones) que manipulan o interac-

³ Dos de los enfoques más populares de TFEA son *Desarrollo conjunto de aplicaciones* (JAD), desarrollado por IBM, y *The METHOD*, desarrollado por Performance Resources, Inc., Falls Church, VA.

tionan con los objetos. Finalmente, se desarrollan también listas de *restricciones* (p. ej.: costes, tamaño, peso) y *criterios de rendimiento* (p. ej.: velocidad, precisión). Se informa a los asistentes que no se espera que las listas sean exhaustivas, pero que sí que reflejen su punto de vista del sistema.

Por ejemplo⁴, imagínese que a un equipo de trabajo TFEA de una compañía de productos para el consumidor se le ha dado la siguiente descripción del producto:

Nuestras investigaciones indican que el mercado de sistemas de seguridad para el hogar está creciendo a un ritmo del 40 por ciento anual. Nos gustaría entrar en este mercado construyendo un sistema de seguridad para el hogar basado en un microprocesador que proteja y/o reconozca varias situaciones indeseables tales como irrupciones ilegales, fuego, inundaciones y otras. El producto, provisionalmente llamado HogarSeguro, utilizará los sensores adecuados para detectar cada situación, puede programarse por el propietario del hogar y llamará automáticamente a una agencia de vigilancia cuando se detecte alguna de estas situaciones.

En realidad, se proporcionaría considerablemente más información en esta fase. Pero incluso con información adicional, habría ambigüedades presentes, existirán omisiones probablemente y podrían ocurrir errores. Por ahora, la «descripción de producto» anterior bastará.

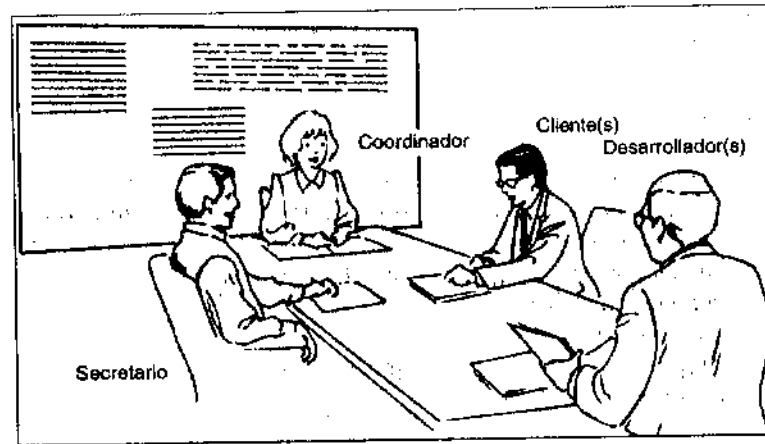
El equipo TFEA está compuesto por representantes de mercadotecnia, ingeniería del software y del hardware y de la fabricación. Participará un coordinador externo.

Todos los componentes del equipo TFEA (Figura 11.2) desarrollan las listas descritas anteriormente. Los objetos descritos para HogarSeguro podrían incluir detectores de humo, sensores de ventanas y puertas, detectores de movimientos, una alarma, un acontecimiento (se ha activado un sensor), un panel de control, una pantalla, números de teléfono, una llamada de teléfono, etc. La lista de servicios podría incluir la instalación de la alarma, vigilancia de los sensores, marcado de teléfono, programación del panel de control y lectura de la pantalla (fíjese que los servicios actúan sobre los objetos). De igual manera, todos los asistentes TFEA desarrollarán una lista de restricciones (p. ej.: el sistema debe tener un coste de fabricación de menos de 200\$, debe tener una interfaz amigable con el usuario y debe conectar directamente con una línea telefónica estándar) y de criterios de rendimiento (p. ej.: un acontecimiento detectado por un sensor debe reconocerse en un segundo; se debería implementar un esquema de prioridad de acontecimientos).

⁴ Este ejemplo (con extensiones y variaciones) se empleará para ilustrar métodos importantes de ingeniería del software en muchos de los capítulos siguientes. Como ejercicio, sería útil que realizara su propia reunión TFEA y desarrollara un conjunto de listas.

FIGURA 11.2

La reunión TFEA



Cuando empieza la reunión, el primer tema de estudio es la necesidad y justificación del nuevo producto, pues todo el mundo debería estar de acuerdo en que el desarrollo (o adquisición) del producto está justificada. Una vez que se ha conseguido el acuerdo, cada participante presenta sus listas para su estudio. Las listas pueden exponerse en las paredes de la habitación usando largas hojas de papel, pinchadas o pegadas, o escritas en una pizarra. Idealmente debería poderse manejar separadamente cada entrada de las listas para poder combinarlas, borrarlas o añadir otras. En esta fase, está estrictamente prohibido el debate o las críticas.

Una vez que se han presentado las listas individuales sobre un tema, el grupo crea una lista combinada. La lista combinada elimina las entradas redundantes y añade las ideas nuevas que se les ocurran durante la presentación, pero no se elimina nada más. Cuando se han creado las listas combinadas de todos los temas, sigue el estudio —moderado por el coordinador—. La lista combinada es ordenada, ampliada o redactada de nuevo para reflejar apropiadamente el producto o sistema que se va a desarrollar. El objetivo es desarrollar una lista de consenso por cada tema (objetos, servicios, restricciones y rendimiento). Después estas listas se ponen a parte para utilizarlas posteriormente.

Una vez que se han completado las listas de consenso, el equipo se divide en subequipos; cada uno trabaja para desarrollar una *mini-especificación*

de una o más entradas de cada lista. La mini-especificación es una elaboración de la palabra o frase contenida en una lista. Por ejemplo, la mini-especificación del objeto **panel de control** de HogarSeguro podría ser:

- montado en la pared;
- tamaño aproximadamente de 9 × 5 pulgadas;
- contiene el teclado estándar de 12 teclas y teclas especiales;
- contiene una pantalla LCD de la forma mostrada en el bosquejo (no presentado aquí);
- toda la interacción del cliente se hace a través de las teclas usadas para activar o desactivar el sistema;
- software para proporcionar una directriz de empleo, recordatorios, etc., conectado a todos los sensores.

Cada subequipo presenta entonces sus mini-especificaciones a todos los asistentes TFEA para estudiarlas. Se realizan nuevos añadidos, eliminaciones y se sigue elaborando. En algunos casos, el desarrollo de algunas mini-especificaciones descubrirá nuevos objetos, servicios, restricciones o requisitos de rendimiento que se añadirán a las listas originales. Durante todas las estudios, el equipo sacará a relucir aspectos que no podrán resolverse durante la reunión. Se hará una lista de estas ideas para tratarlas más adelante.

Una vez que se han completado las mini-especificaciones, cada asistente de la TFEA hace una lista de criterios de validación del producto/sistema y presenta su lista al equipo. Se crea así una lista de consenso de criterios de validación. Finalmente, uno o más participantes (o algún tercero) es asignado para escribir el borrador entero de especificación con todo lo expuesto en la reunión TFEA.

TFEA no es la panacea de los problemas que se encuentran en las primeras reuniones de requisitos, pero el enfoque de equipo proporciona las ventajas de muchos puntos de vista, estudio y refinamiento instantáneo y un paso adelante hacia el desarrollo de una especificación.

11.2.3. Despliegue de la función de calidad

El *despliegue de la función calidad* (DFC) es una técnica de gestión de calidad que traduce las necesidades del cliente en requisitos técnicos del software. Originalmente desarrollado en Japón y usado por primera vez en Kobe Shipyard of Mitsubishi Heavy Industries, Ltd. A primeros de los años 70, DFC «se concentra en maximizar la satisfacción del cliente» [ZUI.92]. Para conseguirlo, DFC hace énfasis en entender lo que resulta valioso para el cliente y después desplegar estos valores a lo largo del proceso de ingeniería.

DFC identifica tres tipos de requisitos [ZUL92]:

Requisitos normales. Se declaran objetivos y metas para un producto o sistema durante las reuniones con el cliente. Si estos requisitos están presentes, el cliente quedará satisfecho. Ejemplos de requisitos normales podrían ser peticiones de tipos de presentación gráfica, funciones específicas del sistema y niveles definidos de rendimiento.

Requisitos esperados. Estos requisitos son implícitos al producto o sistema y pueden ser tan fundamentales que el cliente no los declara explícitamente. Su ausencia sería motivo de una insatisfacción significativa. Ejemplos de requisitos esperados son: facilidad de interacción hombre-máquina, buen funcionamiento y fiabilidad general, y facilidad de instalación del software.

Requisitos innovadores. Estas características van más allá de las expectativas del cliente y suelen ser muy satisfactorias. Por ejemplo, se pide un software procesador de textos con las características estándar. El producto entregado contiene ciertas capacidades de diseño de página que resultan muy válidas y que no eran esperadas.

En realidad, el DFC se extiende a todo el proceso de ingeniería [AKA90]. Sin embargo, muchos conceptos DFC son aplicables al problema de la comunicación con el cliente a que se enfrenta un ingeniero del software durante las primeras fases del análisis de requisitos. Presentamos una visión general sólo de estos conceptos (adaptados al software de computadora) en los párrafos siguientes.

En las reuniones con el cliente el *despliegue de función* se emplea para determinar el valor de cada función requerida para el sistema. El *despliegue de información* identifica tantos los objetos de datos como los acontecimientos que el sistema debe producir y consumir. Estos están unidos a las funciones. Finalmente, la *exposición de tareas* examina el comportamiento del sistema o producto dentro del contexto de su entorno. El *análisis de valor* es llevado a cabo para determinar la prioridad relativa de requisitos determinada durante cada uno de los tres despliegues mencionados anteriormente.

El DFC utiliza observaciones y entrevistas con el cliente, emplea encuestas y examina datos históricos (p. ej.: informes de problemas) como datos de base para la actividad de recogida de requisitos. Estos datos se traducen después en una tabla de requisitos —denominada *tabla de opinión del cliente*— que se revisa con el cliente. Entonces se usa una variedad de diagramas, matrices y métodos de evaluación para extraer los requisitos esperados e intentar obtener requisitos innovadores [BOS91].

11.3. PRINCIPIOS DEL ANÁLISIS

Durante las dos pasadas décadas, los investigadores han identificado los problemas del análisis y sus causas y han desarrollado varias notaciones de modelado y sus correspondientes conjuntos de heurísticas para solucionarlos. Cada método de análisis tiene su punto de vista. Sin embargo, todos los métodos de análisis se relacionan por un conjunto de principios operativos:

1. Debe representarse y entenderse el dominio de información de un problema.
2. Deben definirse las funciones que debe realizar el software.
3. Debe representarse el comportamiento del software (como consecuencia de acontecimientos externos).
4. Deben dividirse los modelos que representan información, función y comportamiento de manera que se descubran los detalles por capas (o jerárquicamente).
5. El proceso de análisis debería ir desde la información esencial hasta el detalle de la implementación.

Aplicando estos principios, el analista se aproxima al problema sistemáticamente. Se examina el dominio de información de manera que pueda entenderse completamente la función. Se emplean modelos para poder comunicar de forma compacta las características de la función y su comportamiento. Se aplica la partición para reducir la complejidad. Son necesarias las visiones esenciales y de implementación del software para acomodar las restricciones lógicas impuestas por los requisitos del procesamiento y las restricciones físicas impuestas por otros elementos del sistema.

Además de los principios operativos de análisis mencionados anteriormente, Davis [DAV95a] sugiere un conjunto⁵ de principios directrices para la «ingeniería de requisitos»:

- *Entender el problema antes de empezar a crear el modelo de análisis.* Hay tendencia a precipitarse en busca de una solución, incluso antes de entender el problema. ¡Esto lleva a menudo a un elegante software para el problema equivocado!
- *Desarrollar prototipos que permitan al usuario entender como será la interacción hombre-máquina.* Como el concepto de calidad del software se basa a menudo en la opinión sobre la «amigabilidad» de la interfaz, el desarrollo de prototipos (y la iteración que se produce) es altamente recomendable.

⁵ Aquí se menciona sólo un pequeño subconjunto de los principios de ingeniería de requisitos de Davis. Para obtener más información, véase [DAV95a].