Ingeniería de Software I

- ✓ Requerimientos
- ✓ Stakeholders y Actores
- ✓ Nuevas técnicas de relevamiento
- ✓ Artefactos
- ✓ Estudio de Factibilidad

Esc. Superior Nº 49 "Cap. Gral. J.J. Urquiza"

Técnico Superior en Desarrollo de Software

Primer Año

Contenido

- Requerimientos
    - Introducción.
    - Definición.
    - **Stakeholders y Actores.**
    - Tipos de requerimientos.
    - **Nuevas técnicas de relevamiento.**
    - ¿Quién los descubre?
    - Pasos para recolectar requerimientos.
    - Oportunidades de mejora. Propuestas.
    - Lista consolidada de requerimientos.
    - Funciones excluidas.
    - Glosario de términos.
    - ¿Por qué es importante detectar, especificar y validar los requerimientos?
- Artefactos.
- Estudio de Factibilidad.

Introducción

- Entender los requerimientos de un problema → tarea difícil.
    - El cliente ¿siempre sabe lo que necesita?
    - Los usuarios finales ¿siempre comprenden las características y funciones que le darán un beneficio?

- Cliente: "*Sé que cree que entiende lo que digo, pero lo que usted no entiende es que lo que digo no es lo que quiero decir."*

- Profesional de sistemas → falla en establecer fundamentos sólidos para un nuevo sistema.
    - registra los requerimientos en forma desordenada.
    - dedica muy poco tiempo en verificarlos.

Definición

Algunas definiciones de requerimiento (también conocido como requisito -para nosotros son sinónimos-).

- Un requerimiento es una característica o una descripción de algo que el sistema debe poseer o hacer con el objeto de satisfacer su propósito.
- Requerimientos → encontrar qué es lo que quiere el cliente y que ha sido apropiadamente documentado y validado por el solicitante.
- Solicitante = stakeholder / actor.

Veamos en detalle que es un Stakeholder y que es un Actor.

Stakeholder

¿Alguien sabe lo que significa?

- Definición:
    - Según el diccionario: accionista o tomador de apuesta.
    - En nuestra profesión: persona que es afectada materialmente por el nuevo sistema que se está desarrollando. También lo llamamos INTERESADO.
- No solo se limita a los usuarios finales del sistema, sino que también incluye al inversionista o financista del proyecto y también al equipo de desarrollo.
- ¿Por qué es importante identificar los stakeholders del futuro sistema? Porque de ellos extraemos y validamos los requerimientos.
- La decisión de desarrollar un sistema por lo general involucra a un montón de gente:
    - inversores (los que pagan)
    - gerencias de diferentes niveles
    - equipo de desarrollo
    - usuarios finales
    - proveedores
    - clientes
    - instituciones gubernamentales
    - autoridades regulatorias

    - etc.

Todos los anteriores son los tipos de stakeholders habituales y conocidos en un proyecto de desarrollo de software.

Importante:

- Identificar los tipos de stakeholder del futuro sistema
- Para cada tipo:
    - determinar él o los stakeholders representantes.
    - determinar las responsabilidades que tendrán para con el futuro sistema.
- Son los dueños del problema y fuente de requerimientos:
    - El involucramiento de los stakeholders es IMPERATIVO.
    - Propiciar un ambiente colaborativo es ESENCIAL.
- ¿Cómo identificar a los tipos de stakeholders? Algunas preguntas que pueden hacerse:
    - ¿Quién se verá afectado por el éxito o fracaso de la nueva solución?
    - ¿Quiénes son los usuarios del sistema?
    - ¿Quién es el comprador del sistema?
    - ¿Quién es el sponsor del desarrollo?
    - ¿Quiénes se verán más afectados por el resultado final que produce el sistema?
    - ¿Quién evaluará y firmará la aprobación del sistema cuando este se termine y se implemente?
    - ¿Hay otros usuarios internos o externos del sistema cuyas necesidades deben tomarse en cuenta?
    - ¿Hay cuerpos regulatorios o estándares organizacionales los cuales el sistema deberá obedecer?
    - ¿Quién desarrollará el sistema?
    - ¿Quién instalará y mantendrá el nuevo sistema?
    - ¿Quién tendrá actividades de soporte y suministrará entrenamiento para el nuevo sistema?
    - ¿Quién testeará y certificará el nuevo sistema?
    - ¿Quién venderá y pondrá en el mercado al nuevo sistema?
    - ¿Hay alguien más?
- Conocer las competencias de los stakeholders también es importante:
    - habilidades necesarias para realizar su trabajo
    - nivel de conocimiento del problema
    - calificaciones en el negocio
    - nivel de conocimientos en computación
    - otras aplicaciones que usan
- ¿Cómo interactuar e involucrar a los stakeholders? → con las técnicas y métodos de relevamiento visto en clases anteriores.
- ¿Qué tenemos que lograr involucrando a los stakeholders? → entender sus necesidades.

Necesidad → Un reflejo del problema (u oportunidad de mejora) del negocio, que debe ser encarado para justificar el desarrollo, la compra o el uso de un nuevo sistema. Esta es la fuente para detectar los **requerimientos**.

Actor

- ¿Qué es un actor? → Es un stakeholder que además tendrá interacción con el sistema. Es decir, va a hacer uso del sistema.
- ¿Para qué un actor usa un sistema? → porque necesitan cumplir objetivos o metas.
- ¿Por qué es importante identificarlos?
  - son los dueños de los objetivos o metas que el nuevo sistema tiene que satisfacer.
  - permiten además establecer los límites del sistema (de qué será responsable y de qué no será responsable).
- Importante tener presente que:
  - los actores no son siempre personas.
  - no hay que confundir a los actores con los dispositivos que ellos usan.
  - no confundir actores con roles organizacionales o cargos laborales.

Lista de actores y stakeholders:

Uno de los primeros 'artefactos' que comenzamos a crear.
Es una tabla de 3 columnas: Nombre, Descripción, Tipo (A o S) (**A**ctor o **S**takeholder) y Categoría (Primario, Iniciador, Secundario, Terciario).

Nombre	Descripción	Tipo	Categoría

Categorías de Actores:

Clasificación	Descripción
Primario	Dueño de la meta del caso de uso, con o sin interacción directa según sea el caso.
Iniciador	Actor que inicia físicamente el caso de uso (elegir un punto del menú, presionar un botón, presionar una tecla o iniciar la historia de alguna manera). En los Casos de Uso Resumen es el iniciador del primer paso.
Secundario	Actor que tiene interacción directa con el caso de uso y que no es ni actor primario, ni es actor iniciador.
Terciario	Stakeholder (interesado) que no es actor primario, que no tiene interacción directa con el caso de uso pero sí tiene interés, y que provee o es destinatario de algún dato o información.

Nota 1: Que un actor tenga interacción directa significa que ingresa datos o recibe informes (comprobantes, comunicaciones, etc.) por distintas vías (monitor, impresora, celular, e-mail, etc.) cuyo formato de interfaz de usuario está destinado a él.
Nota 2: Un actor puede ser primario e iniciador, o sea pertenecer a las dos categorías, si es el dueño de la meta e inicia físicamente el caso de uso.
Nota 3: Todos los actores son stakeholders pero no todos los stakeholders son actores.

Continuando con los requerimientos …

Importante:
- No es lo mismo un pedido o un deseo o una necesidad o un problema que un requerimiento.
- No todos los pedidos o deseos o necesidades o problemas se convierten en requerimientos, pero un requerimiento se origina a partir de ellos.
- Para que se conviertan en requerimiento deben ser **documentados** y **validados**.

Tipos de requerimiento

- **Funcionales**:
    - Describen una interacción entre el sistema y su ambiente. Los requerimientos funcionales describen cómo debe comportarse el sistema ante un estímulo.
    - Ejemplo → Sistema de control de alumnado en el Terciario Urquiza.
      Requerimientos funcionales:
        - Inscribir alumno a carrera.
        - Inscribir alumno a mesa de examen.
        - Informar listado de materias para un año determinado de una carrera.
        - Emitir certificado analítico de materias rendidas para un alumno.
    - Funcionales temporales:
        - En algunos sistemas ciertas actividades deben realizarse en ciertos momentos. Ejemplo: El primer día hábil de cada mes, generar la liquidación de horas extras del mes anterior.

- **No funcionales**:
    - Un requerimiento no funcional es una restricción sobre el sistema o su proceso de producción.
    - Algunos ejemplos de requerimientos no funcionales son los siguientes:
        - El sistema debe ejecutar sobre Linux Ubuntu.
        - El sistema debe ser desarrollado íntegramente en software libre.
        - El sistema debe ser adaptable a dispositivos móviles.
        - El sistema debe impedir que usuarios no autorizados accedan a información crítica, implementando un esquema de seguridad de roles de usuario.
        - El sistema debe estar en funcionamiento 24*7.
        - El sistema debe poder procesar 100.000 transacciones por hora.
        - El sistema debe ser desarrollado en HTML5, CSS3, JavaScript, PHP y MySQL.
    - Un problema recurrente → sólo tener en cuenta los requerimientos funcionales.

Los sistemas sufren modificaciones en etapas avanzadas del proceso de desarrollo porque no se tuvieron en cuenta estos requerimientos → mayores costos, demora en la entrega final, mucho riesgo de retrabajo.

¿Quién descubre los requerimientos?

- TODOS los stakeholders, incluyendo al equipo de desarrollo.

- Detectar los stakeholders y actores, involucrarlos en el proyecto, utilizar métodos y técnicas de relevamiento para explicitar y documentar los requerimientos.

Pasos a seguir para recolectar requerimientos:

- Encontrar, comunicar, recordar o documentar lo que se necesita realmente → significado claro para los stakeholders y los miembros del equipo de desarrollo.
- Comprensión de los requerimientos → concepción, para definir el alcance y la naturaleza del problema a resolver.
- Indagación → ayuda a los stakeholders a definir lo que se requiere.
- Documentación.
- Validación.

Oportunidades de mejora:

- Es importante destacar que el nuevo sistema debe ser una oportunidad para proponer mejoras en el proceso de negocio del cliente.

Proponer → pueden haber más de una oportunidad de mejora. El cliente es el que tendrá la última palabra.

Para encontrar, comprender e indagar los requerimientos, mencionamos nuevas técnicas de relevamiento, utilizadas frecuentemente por el equipo de desarrollo de software, y que son complementarias a las vistas en apuntes anteriores (entrevista, cuestionario, muestreo, observación y recolección de datos reales).

Talleres de requerimientos:

El taller de requerimientos está diseñado para fomentar un consentimiento sobre los requerimientos de la aplicación y obtener un rápido acuerdo sobre un curso de la acción, en un corto tiempo.
Beneficios:

- Ayuda en la creación de un equipo eficaz, en donde todos los interesados opinan y nadie queda afuera.
- Genera un acuerdo entre las partes interesadas y el equipo de desarrollo en cuanto a que es lo que debe hacer la solicitud.
- Se pueden exponer y resolver cuestiones políticas que interfieren con el éxito del proyecto.

<u>Pasos para desarrollar un taller de requerimiento:</u>

- Identificar y/o verificar quiénes son los patrocinadores, tomadores de decisión, y usuarios representativos.
- Organizar la forma en que se llevará a cabo el taller y asegurar como los involucrados participarán en el mismo.
- Aplicar diferentes técnicas (Por ej. la lluvia de ideas, juego de roles, etc.) para desarrollar y captar las necesidades del negocio, con el control de un moderador experimentado.
- Una vez aseguradas las necesidades de negocio se deberá realizar una reducción de ideas, para luego ordenarlas de forma jerárquica para definir los requerimientos más importantes.

Tormenta o Lluvia de ideas o Brainstorming:

Es una técnica grupal que permite indagar u obtener información acerca de lo que un grupo conoce sobre un tema determinado. La idea es que cada participante haga aportaciones.

Características:

- Se posee un tema o pregunta central.
- La participación puede ser oral o escrita.
- Debe existir un mediador (moderador).
- Participa un grupo reducido de personas.

Juegos de Roles:

Es una sesión en la que el moderador organiza un escenario en el que se asignan diferentes papeles a los participantes, papeles que se identifican con los de la situación en la que estos se encontrarán cuando comiencen su trabajo.

Objetivos:

- El juego de roles da a los participantes oportunidades de ejecutar varios roles que representarán papeles reales que encontrarán en el trabajo verdadero.
- Un resultado importante es que los participantes tienen la oportunidad de ver las situaciones desde perspectivas diferentes a la que tendrían en la realidad.

Esta técnica es útil para manejar aspectos o temas difíciles en los que es necesario tomar diferentes posiciones para su mejor compresión. Resultan muy útiles en las revisiones anuales o semestrales de los diferentes programas.

Revisiones:

Las revisiones son reuniones formales o informales cuyo objetivo es revisar algo para detectar y corregir errores. Por ejemplo: un documento, un prototipo, el sistema en desarrollo, etc.

Las organizaciones podrán definir también:

- Requisitos del producto.
- Resolución de eventuales requisitos del contrato o del pedido.
- Capacidad de la empresa para satisfacer tales requisitos.

Con estas técnicas de relevamiento vistas, más las anteriores, el objetivo es confeccionar otro artefacto denominado **Minutas de relevamiento.**

Las minutas son el recurso escrito de una reunión o audiencia.

Es un artefacto mediante el cual se deja registro de las conclusiones obtenidas.

Proporcionan una descripción de la estructura de la reunión, comenzando con una lista, siguiendo con los planteamientos y las respuestas de cada uno de los asistentes, finalizando con las conclusiones arribadas.

La minuta debe ser:

- Específica. Se deben evitar abstracciones y generalidades.
- Concisa. La información debe ser, preferentemente, breve y puntual.
- Objetiva. Es necesario eliminar toda subjetividad, a la hora de formular la minuta.

Pasos a seguir:

- Anotar el lugar y la fecha de la reunión.
- Tomar lista de los participantes de la reunión y circular lista de asistencia.
- Listar cada tema, en el orden que se fueron tratando.
- Clasificar los temas fundamentales, dejando en segundo plano los menos importantes.
- Anotar actividades que deben realizarse hasta la próxima reunión para el cumplimiento de cada uno de los temas.
- Anotar la fecha y lugar en la que se efectuará la siguiente reunión.
- Si no se construye durante el relevamiento en forma colaborativa, debe ser enviada a posteriori a todos los involucrados para consenso.
- Una vez consensuada por todos, se transforma en una herramienta de trabajo para el análisis.

Secciones:

- Encabezado
- Cuerpo
- Pie

Organización propuesta:

- Estructura organizativa y descripción del negocio o de los procesos de negocio actuales
- Información complementaria.
- Necesidades para con el nuevo sistema.
- Cuestiones abiertas.
- Historial de versiones

Retomamos y continuamos con los requerimientos …

A medida que vamos detectando los requerimientos tenemos que ir armando un nuevo artefacto, la **Lista consolidada de requerimientos**:

- Consolida, organiza y estructura la información recolectada durante la etapa de obtención de requerimientos.
- La lista va a ir evolucionando con el tiempo, en la medida que crezca en volumen nos veremos en la necesidad de organizarla por tipos de requerimientos, actor, meta u objetos. Los criterios dependerán en gran medida de las necesidades del analista para administrar la lista.
- Vocabulario y notaciones utilizadas → deben ser estrictamente las del cliente o las del dominio de aplicación.
- **NO** debería incluir:
  - tabla, registro, campo, lista enlazada, función, proceso o procedimiento (en su acepción informática), variable, base de datos, archivo, directorio, sistema operativo, módulo, objeto, clase, hilo de control, servidor, HTTP, webservice, etc.

    - "el sistema deberá implementar un módulo..." o "un webservice tomará los datos de la tabla personas y los enviará vía SOAP al servidor Web..."
- **SI** podría incluir:
    - "el sistema deberá implementarse en C++..." o "el sistema deberá procesar 10.000 transacciones por minuto..."
- Dividida en secciones → para agrupar requerimientos relacionados con funcionalidades específicas del nuevo sistema.
- Cada entrada en la lista debe
    - describir un requerimiento individual, particular o específico, puntual.
    - respetar los atributos de calidad de un requerimiento → cohesión y observable externamente.
- Ejemplo de un requerimiento no cohesivo:
    - El sistema debe soportar la generación y control de objetos de configuración; es decir, objetos que son agrupaciones de otros objetos del sistema. El sistema de control de configuraciones deberá permitir el acceso a objetos en un grupo utilizando incluso un nombre incompleto.
    - Observaciones:
        - se define el concepto de objetos de configuración → "Glosario de términos".
        - hay un requerimiento general y un requerimiento muy detallado y específico → lo único a incluir en esta sección.
- Formato de cada ítem de la lista consolidada de requerimientos:
    - Identificador
    - Descripción → lenguaje natural o con notación entendible para el cliente
        - NO incluir "Si", "Cuándo", "pero", "excepto", "a menos" y "aunque".
        - EVITAR "usualmente", "generalmente", "a menudo", "normalmente" y "típicamente".
        - Usar lenguaje no especulativo.
        - Formato sugerido: Acción/es + objeto/s + (para qué) + (para quién) + comentarios imprescindibles.
          Verbos sugeridos para 'Acción': Registrar, actualizar, informar, listar, validar, calcular, generar, publicar, borrar, habilitar, etc.
          Usaremos: **Validar** para los controles que realiza el Sistema.
          Controlar, para los controles que realizan las personas.
        - Ejemplos:
            - Informar las ventas mensuales para decidir cambios de precio al Gerente de ventas.
            - Registrar reserva del cliente.
            - Informar las ventas mensuales para decidir cambios de precio al Gerente de ventas, de los productos cuyo volumen de venta haya superado en un 50% los niveles del mes anterior.
    - Tipo (Funcional o No Funcional)
    - Prioridad (alta/media/baja o escala numérica).
    - Otros atributos, como por ejemplo el autor o responsable del requerimiento.

Identificador	Descripción	Tipo	Prioridad

Ciclo de vida de la lista de requerimientos

La lista de requerimientos es un artefacto que comienza en la fase de inicio, podríamos decir que nace como un lista desordenada, desnivelada y hasta ambigua. En la medida que el relevamiento se desarrolla los requerimientos evolucionan, cambian, se ajustan, algunos aparecen de la nada y otros pierden fuerza hasta desaparecer o transformarse en otra cosa. La lista de requerimientos es un artefacto de trabajo que se ampliará y contraerá durante algún tiempo hasta estabilizarse.

No es factible impedir a los stakeholder que piensen, pues en la medida que avanza el proceso ellos descubren cosas tanto como los analistas. Es un proceso de simbiosis entre el cliente y el analista, y está bien que así suceda. Lo adverso sería que compitieron por los requerimientos rechazándose hasta destruir la relación. Esto no significa que aceptemos cambios todo el tiempo y tampoco que los negaremos sin por lo menos analizar su trascendencia. El equilibrio será lo más difícil de hallar pero no imposible.

La lista terminará siendo un documento mucho más complejo que una lista y quizás esté particionado en varias sub-listas para hacer posible su entendimiento por todos aquellos que la usen. Un sistema pequeño tendrá docenas de requerimientos. Hacer una lista de ellos no será fácil de administrar. El contexto y el ingenio del analista sumado a su experiencia ayudarán.

Para completar la especificación inicial de un sistema, podríamos utilizar dos artefactos más: una lista de funciones excluidas y un glosario.

Funciones excluidas:

- La lista consolidada de requerimientos permite definir el límite del sistema → lo que deberá satisfacer.
- Muchas veces es igual de importante, dejar en claro lo que el sistema NO se hará responsable.
    - debe quedar claro para todos los stakeholders.
    - debe ser validada por los stakeholders.

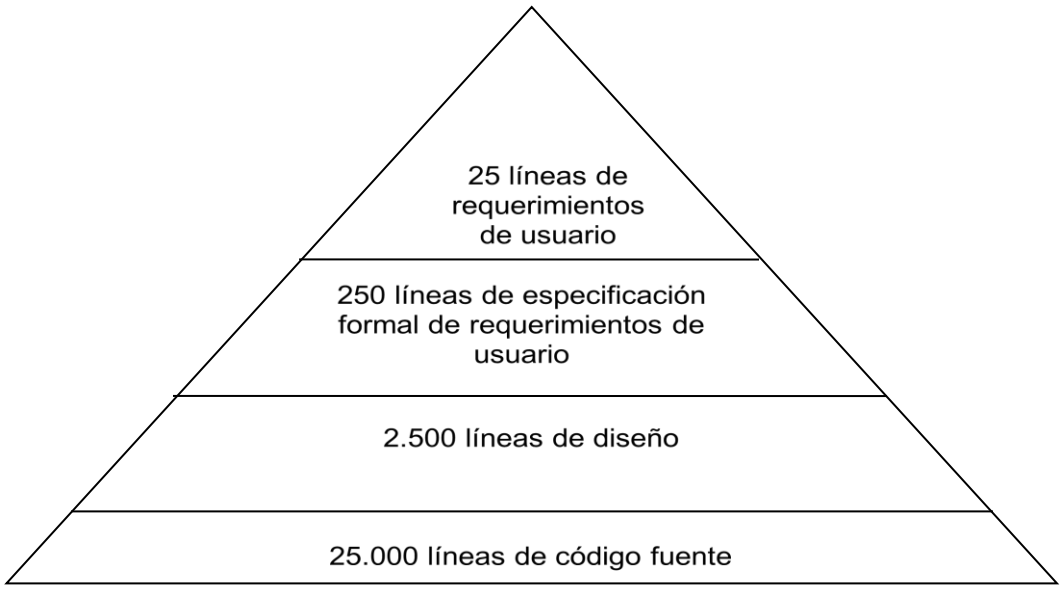
Debe ser redactada teniendo en cuenta las mismas consideraciones que para la lista consolidada de requerimientos.

Glosario de términos:

- Artefacto que permite definir cada uno de los términos de negocio del cliente, o propios del dominio del nuevo sistema.
- Se da una definición clara de cada uno de estos términos y llegado el caso se pueden adjuntar sinónimos.

¿Por qué es importante detectar y especificar los requerimientos adecuadamente?

Algunos estudios han logrado establecer la siguiente relación:



- Algunos casos:
  - Aeropuerto de Denver:
    - 1993
    - Sistema subterráneo de traslado de equipaje: 4.000 telecarros independientes
    - Software para el control del sistema de carros (21 meses)
    - La inauguración se debió postergar 3 veces
    - El presupuesto era de USD 193M
    - BAE Automated Systems reconoció que no podía predecir el momento en que lograría estabilizarlo.
    - En 2005 United decidió abandonar el sistema y ahorrarse USD 1M por mes en mantenimiento

  - Satélite Clementine:
    - Proyecto conjunto entre Departamento de Defensa (EE.UU.) y NASA
    - Parte del programa de defensa denominado Guerra de las Galaxias
    - Satélite para selección de blancos
    - Por un error en el software de control, en lugar de situar a la Luna en la mira, el sistema encendió los motores durante 11 minutos y agotó el combustible.

  - Encuesta de IBM sobre sistemas distribuidos (aprox. 1994):
    - 55% costó más de lo calculado
    - 68% no cumplió los plazos
    - 88% tuvo que re-diseñarse casi desde cero

- Ariane-5:
  - Hito en el proyecto espacial europeo;
  - Terminó en desastre debido a una falla en el software que controlaba el movimiento vertical.

- Therac-25 (radioterapia):
  - http://es.wikipedia.org/wiki/Therac-25
  - Dispositivo médico de radioterapia para el tratamiento del cáncer.
  - Se reemplazaron ciertos controles de hardware por software.
  - El software falló y causó la muerte de varias personas por sobredosis.

- FBI:
  - Gastó 170 millones de dólares en el Virtual Case File para luego abandonarlo completamente.

- Administración Federal de Aviación - EEUU:
  - Canceló un proyecto para actualizar los sistemas de control aéreo cuando ya se habían gastado 2.600 millones de dólares.
  - FoxMayer Drug Co - EEUU:
  - Se fundió luego de que su ERP que costó USD 40M mostrara innumerables fallas.

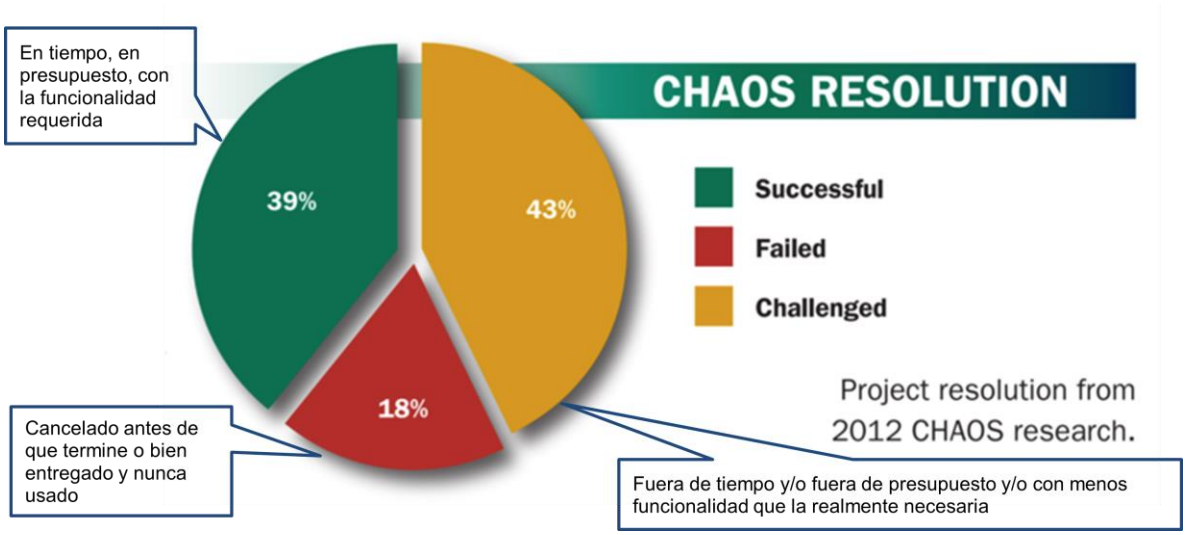
- Bank of America:
  - En la década del 80 perdió USD 80M y todo su negocio de trusts a raíz de la imposibilidad de terminar el sistema para administrarlos, conocido como MasterNet.
  - Hasta ese momento era considerado una de las empresas líderes en la adopción de nuevas tecnologías.

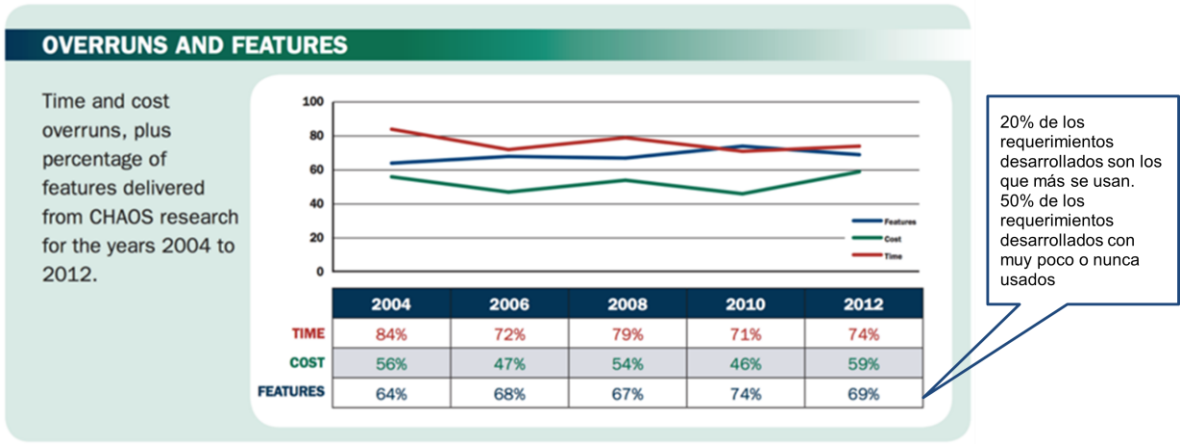
Principales fallas de los proyectos de software:

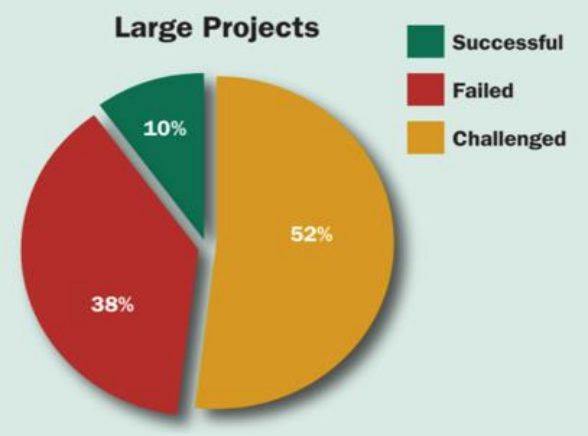
Fracaso:

- En expectativos
- En costo
- En calidad
- En tiempo

Algunos gráficos que explican las razones del fracaso:







Para terminar, una vez que relevamos, detectamos, documentamos y validamos los requerimientos, tenemos que analizar la factibilidad de su implementación en el proyecto de desarrollo de software que estamos analizando.

FACTIBILIDAD

El éxito o la probabilidad de éxito de un proyecto se determina por el grado de factibilidad del mismo.

Es una evaluación Cuantitativa y Cualitativa.

Es el proceso para medir aspectos esenciales para el éxito del proyecto y del producto y sus beneficios/ganancias.

Para desarrollar un software tenemos Recursos:

- Económicos.
- Técnicos.
- Operativos.

Por lo tanto, a través de un estudio de factibilidad, debemos analizar la factibilidad:

- Operativa.
- Técnica.
- Económica.
- y otras.

Con este estudio tenemos que responder a la pregunta: **¿Continuamos o no con el proyecto?**

Factibilidad Operativa

- La solución es adecuada para la organización?
- Es realmente lo que necesita la organización?
- Vale la pena su resolución?
- Funcionará?
- Qué opina la Gerencia y los usuarios finales?

- Qué tanto es el deseo del cambio expresado por los usuarios y el personal involucrado.
- Hay miedo al cambio – reemplazo hombre /máquina?
- Los cambios en las prácticas organizativas pueden provocar una obsolescencia rápida del sistema?
- Cuál es el nivel de complejidad del nuevo sistema? Si es muy complejo en relación al existente probablemente fracase.
- Y finalmente, que tan posible es terminar el producto final con las condiciones actuales, en base a estimaciones previas?

Factibilidad Técnica

Evaluación de las TI existente en la organización y recolectar información sobre su posibilidad de uso y evaluar los que deben ser adquiridos.

- Están disponibles los recursos técnicos?
- Es posible conseguirlos?
- Se ajusta a cronograma?
- Está disponible la tecnología para implementar la solución sugerida?
- Es fácilmente aplicable la solución?
- Se cuenta con el conocimiento técnico necesario?
- Es práctica?
- El cliente la puede/quiere aceptar?
- Aspectos referidos a la implementación técnica. Que hard o soft está a nuestro alcance?
- Comunicaciones? Velocidad y calidad.
- Interfaces con otros sistemas. Son posibles?

Factibilidad Económica

- Vamos a obtener ganancias/beneficios?
- Que tanto? Relación costo/beneficio.
- En qué plazo?
- Que tan viable financieramente es este proyecto?
- Costos: de desarrollo – de operación – gastos.
- Beneficios: tangibles – intangibles.
- Fuentes de inversión.

Otras factibilidades

- De tiempo.
- Legales.
- Ambiental.
- etc.

El resultado de esta evaluación debe quedar reflejado en el artefacto **Estudio de Factibilidad** y como dijimos, su resultado determina la continuidad o no del proyecto.

Artefactos

Ya mencionamos varias veces el nombre de artefacto. ¿Qué es?

Un artefacto es un producto tangible resultante del proceso de desarrollo de software.

- Algunos artefactos ayudan a la descripción de la función, la arquitectura o el diseño del software (los veremos más adelante, como los casos de uso, diagrama de clases u otros modelos UML). Otros se enfocan en el proceso de desarrollo en sí mismo, como planes de proyecto, casos de negocios o análisis de riesgos.
- El código fuente compilado para el testeo se suele considerar un artefacto, ya que el ejecutable es necesario para el plan de testeo.

En ocasiones un artefacto puede referirse a un producto terminado, como el código fuente o el ejecutable, pero más habitualmente se refiere a la documentación generada a lo largo del desarrollo del producto en lugar del producto en sí.

Los artefactos pueden variar en su necesidad de mantenimiento y actualización.

Los artefactos que detallan el diseño pretendido para el producto suelen realizarse al principio del proyecto y no necesitan mantenerse, mientras que otros se mantienen a lo largo del ciclo de vida con información que se actualiza durante el desarrollo.

Los artefactos vistos hasta ahora y que se deberían incluir en la especificación de un sistema son:

- **Minutas de relevamiento.**
- **Cursogramas.**
- **Tabla de stakeholders / actores.**
- **Lista consolidada de requerimientos.**
- **Funciones excluidas.**
- **Glosario.**
- **Estudio de factibilidad.**

Seguiremos viendo más artefactos para modelar y especificar un sistema … estos van a especificar cada vez con mayor detalle el sistema a construir.