UNIVERIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CONTADURÍA Y ADMINISTRACIÓN

PROPUESTA PARA MEJORAR EL DESARROLLO DE SOFTWARE EN LA JEFATURA DE UNA EMPRESA

DISEÑO DE UN PROYECTO PARA UNA ORGANIZACIÓN
QUE PARA OBTENER EL TÍTULO DE:

LICENCIADO EN INFORMÁTICA

PRESENTA:

JOSÉ SERGIO PAZ LUCAS

ASESOR:

M. I. MARÍA ISABEL GARRIDO GALINDO

MÉXICO, D.F. 2006

UNIVERIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CONTADURÍA Y ADMINISTRACIÓN

PROPUESTA PARA MEJORAR EL DESARROLLO DE SOFTWARE EN LA JEFATURA DE UNA EMPRESA

DISEÑO DE UN PROYECTO PARA UNA ORGANIZACIÓN

JOSÉ SERGIO PAZ LUCAS

MÉXICO, D.F. 2006

Índice

Capítulo I Introducción

Este proyecto presenta una propuesta para mejorar el desarrollo de software con base en algunas de las mejores prácticas que hay en la industria del software. El planteamiento de la propuesta se encuentra elaborado conforme a las características que en un momento determinado presenta una jefatura del área de sistemas de una empresa privada. Con base a esas características particulares se seleccionaron y ajustaron los elementos que integran la propuesta.

Las mejores prácticas son un conjunto de experiencias que se han recolectado a través del tiempo por varias organizaciones para proporcionar puntos de referencias acerca de posibles soluciones u oportunidades experimentadas. Las mejores prácticas que son abordadas en este proyecto tienen un reconocimiento por parte de la industria del software por los beneficios que han obtenido al utilizarlas. Cada una de ellas tiene un enfoque diferente y para aplicarlas en un caso particular como este proyecto es importante seleccionar las partes adecuadas que permitan alcanzar los objetivos planteados. Cabe resaltar que el proyecto representa para la jefatura un primer paso para mejorar la calidad en el desarrollo de software. Esta situación no es exclusiva de una organización, sino que es compartida por muchas de las empresas en México. La competencia local y global exige mayores niveles de calidad para satisfacer las exigencias del mercado y para lograrlo no existe un camino establecido infalible, en cambio existen varias opciones.

El nombre real de la empresa y cualquier elemento que forme parte de ella no son mencionados por solicitud expresa de la misma empresa.

Los estándares y modelos consultados no se encuentran contenidos por completo dentro de este documento ya que son empleados de manera parcial, por lo que se hace una breve descripción y algunas referencias bibliográficas para que se tenga una idea general de lo que trata cada uno.

Debido a la naturaleza de este documento no se cuestionan los fundamentos de las mejores prácticas empleadas, ya que para realizarlo se requiere de un estudio más profundo del que se está presentando. Es por esto que la perspectiva crítica será postergada para que pueda ser desarrollada en un documento que cuente con las características necesarias para realizarlo.

Preámbulo

Es indudable la presencia de la tecnología y su intervención en la vida diaria; nos proporciona apoyo en tareas que pasan desde las vanas hasta aquellas en las que se involucran vidas humanas. Cualquier mal funcionamiento trae consigo resultados indeseados que dependerán de la función que desempeñan. Como parte de la ética profesional es inexorable la responsabilidad de los integrantes involucrados de entregar productos que funcionen bien pese al esfuerzo que se requiere para hacerlo posible.

Existen ideas que son persistentes en nuestro momento histórico (utilidad, progreso, eficiencia, eficacia, técnica, ciencia, etc.) que nos permiten conformar una plataforma para resolver problemas y establecer prioridades. Aunque el nombre de estas ideas se mantiene con el tiempo el significado variará en muchos de los casos dependiendo de las circunstancias. Cualquier actividad o producto que pretenda desarrollarse debe participar de alguna forma en esta plataforma para que sea justificada y genere interés dentro de este entramado de relaciones. Uno de los artificios más notorios dentro del contexto actual es el software. El software ha posibilitado la concreción, en cierta forma, de muchas de las ideas y problemas que se han planteado, pero también ha mostrado que su aplicación puede tener un impacto incierto y a veces negativo. Esto último provocado por la incapacidad para abordar todas las consecuencias que puede tiene el software u otros temas que por su generalidad y poca claridad son relegados a problemas de segundo plano con menor prioridad a comparación de los problemas más concretos como los tiempos de entrega,

costos, funcionalidad, etc. Pues más que ser un problema técnico es un problema de carácter humano lo que esto implica, ya que es el hombre quien construye, utiliza y se ve afectado por el uso de productos de software, por lo que la visión del desarrollo de software debe de ampliar su visión para plantear este tipo de problemas que le incumben por ser un elemento que participa de manera importante en la actualidad. Y aunque en este proyecto se plantean problemas particulares nunca está por demás denotar la importancia de cuestiones más generales.

Problemática

En este caso en particular nos ocupa el desarrollo de software de una empresa que su giro no es el desarrollo de software y solamente se apoya en él para sus actividades primarias. El desarrollo de software es una actividad que provee un servicio que pretende satisfacer necesidades específicas. Este proyecto surge con la intención de responder a las siguientes preguntas ¿cómo mejorar el software que se desarrolla?, ¿cuál es la mejor manera de desarrollar software? y ¿cómo asegurar que el software que se desarrolla realmente contribuye al logro de los objetivos estratégicos de la organización? Es inevitable pensar que las respuestas no se tornarán fáciles ya que no implica solamente cambiar una pieza por otra, como si fuese un proceso mecánico, sino que en ella interviene el factor humano, lo que le agrega cierto grado de complejidad. Por ejemplo, una persona cuenta con hábitos de trabajo y muchas veces es renuente a abandonarlos, tratar de convencerla de aplicar nuevas formas de realizar su trabajar requiere recursos, paciencia y labor de convencimiento para lograrlo y aunque existen muchas propuestas y modelos que prometen grandes resultados, lo que ocurre en la realidad es que no es posible aplicarlos en todas las organizaciones por igual, lo que puede llevarnos a elegir entre ajustarse al modelo o ajustar el modelo a la organización. Es por esto que es crucial el conocimiento de la situación y el entorno actual en el que se realiza la propuesta para el establecimiento de mejores prácticas para el desarrollo de software.

Cada organización es diferente y tiene sus propias cuestiones por resolver. Estas cuestiones pueden manejarse por prioridades de la siguiente manera: como un problema que requiere solución inmediata, como una necesidad para mejora un elemento de manera planeada o como una oportunidad que requiere un análisis detallado a mediado o largo plazo. Lo cierto es que las organizaciones expresan cuestiones que demandan una solución. A partir de una definición pueden existir múltiples soluciones, pero la organización también postula ciertas restricciones que acotan significativamente el universo de soluciones.

Para la organización objeto del proyecto las características que se consideraron más importantes para definir el entorno sobre el cual se desarrolla el proyecto son las siguientes:

1. El proyecto se enfoca en una jefatura que pertenecen al área de sistemas.
2. La jefatura cuenta con varios proyectos de desarrollo de software y cada uno de ellos con diferentes características.
3. Se tienen asignados quince proyectos hasta ese momento.
4. El equipo de trabajo cuenta con doce integrantes. Los cuales se encuentran distribuidos en cada uno de los proyectos, que pueden desempeñar uno o más roles[1].
5. Los sistemas presentan algún tipo y grado de dependencia entre sí e incluso depender de otros que en ocasiones que se encuentran fuera de la jefatura.
6. El presupuesto para las actividades de calidad es reducido.
7. No se utiliza software libre.
8. Las actividades de calidad en el desarrollo de software son nuevas dentro de la organización.

[1] Vea el Anexo para ver los roles y la distribución de los proyectos.

9. No hay un equipo dedicado a las actividades de calidad para toda el área de sistemas, sino que cada jefatura lo maneja estas actividades como mejor le parece.
10. Todos los usuarios y clientes son internos a la empresa.
11. Existen cambios constantes en los requerimientos debido a la dinámica del negocio y la ausencia de una manera formal de solicitarlos o modificarlos.
12. Los ciclos de desarrollo son generalmente cortos (3 semanas aproximadamente).
13. El manejo de la seguridad de las aplicaciones, la infraestructura y soporte técnico son manejadas por áreas distintas.

Ante el entorno descrito, surgen gran cantidad de problemas que requieren solución. Problemas que se han manifestado tanto de manera directa en toda el área de sistemas como de manera específica en la jefatura y que difícilmente pueden pasar desapercibidos en las actividades cotidianas.

A continuación se enlistan los problemas más notorios que pudieron identificarse al inicio de este proyecto en lo que se refiere al desarrollo de software de software:

1. No hay manera de cuantificar el impacto de las modificaciones que solicita el usuario al momento de solicitarlas.
2. La metodología de desarrollo propia de la organización se encuentra incompleta y sólo se utilizan ciertas partes.
3. Existen nuevos requerimientos que no requieren llenar una gran cantidad de formatos y pasos como lo marca la metodología existente.
4. Muchos proyectos sobrepasan el tiempo estimado y los costos.
5. No se generan datos históricos que permitan comprender la forma en que se está desarrollando el software.
6. No hay retroalimentación entre el equipo de desarrollo al final de cada ciclo de desarrollo.
7. Las pruebas desarrolladas por las mismas personas que construyen el software no resultan lo suficientemente efectivas.
8. No se cuenta con herramientas automatizadas para realizar pruebas.
9. No se aplican muchas pruebas de manera formal debido a la falta de tiempo, iniciativas, técnicas, poco presupuesto.
10. Las actividades de calidad son vistas como obstáculos porque retrasan los tiempos de entrega.
11. No se encuentra definido un plan que permita mejorar paulatinamente el software que se desarrolla.
12. Las actividades se priorizan en función de los problemas que se presentan.
13. Las modificaciones que solicita el usuario es generalmente para agregar cosas y algunas veces para corregir lo que ya se hizo.
14. Un equipo de desarrollo se encarga de varios proyectos, lo cual hace en ocasiones difícil atenderlos de manera simultánea cuando no se tienen planificados los proyectos.
15. No hay una relación bien definida entre los roles y las tareas que deben desempeñar.
16. Se cuenta con una buena disposición para mejorar la calidad del software que se está desarrollando, pero se desean resultados rápidos y evidentes.
17. Las pruebas que se realizan son manuales y enfocadas principalmente a la funcionalidad con código ejecutable.
18. No existe una administración de la configuración de software.
19. En el caso de algunos sistemas que se desarrollan se tiene que definir también el proceso del negocio porque no se encuentra documentado.
20. Algunas veces los usuarios se niegan a utilizar el software desarrollado porque suele interferir con ciertos intereses creados.
21. El mal funcionamiento del software en producción afecta no sólo al usuario final, sino también a otros sistemas que dependen de él.
22. No hay una cultura de administración de proyectos al igual que de mejora continua.
23. La resistencia al cambio se debe a que el equipo de trabajo no le ve las ventajas significativas a algo que no es tan evidente.

24. Durante la construcción es común que se caiga nuevamente en defectos anteriormente detectados y solucionados.
25. Muchas veces el usuario no sabe lo que quiere y no se cuenta con medios dentro de la organización para ayudarlo.
26. Los canales de comunicación con el usuario no se encuentran del todo definidos.
27. No existen mecanismos que ayuden a mejorar el servicio proporcionado.
28. Existen dependencias hacia las personas que desarrollaron la aplicación para realizar cualquier modificación o mejora a éste.
29. La ausencia de documentación durante el desarrollo de software provoca que ocasionalmente se vuelva a desarrollar nuevamente una parte por la imposibilidad de comprender el trabajo realizado.

Con este panorama, es indispensable elaborar un plan que permita atacar por prioridades cada uno de los problemas. Elaborar un plan de calidad permitirá abordar de manera gradual un conjunto de problemas dependiendo de la prioridad. Esta prioridad dependerá en gran medida del vínculo que se tiene con las estrategias del negocio. Este proyecto es parte de esta fase inicial del plan de calidad.

De la lista de problemas presentada se seleccionaron aquellos que tenían relación directa sobre la estrategia de la empresa y a manera de resumen se conjuntaron en los siguientes tres puntos a resolver por este proyecto:

- Organizar el proceso de desarrollo de software.
- Entregar productos con un funcionamiento adecuado para el usuario.
- Contar con un mecanismo de apoyo en la administración de cada proyecto.

Las soluciones que se postulan para solucionar estos puntos son:

Contar con un orden en el desarrollo de software. Para esto contamos con diferentes modelos de referencia con o sin procesos incluidos y su alcance depende de la perspectiva que tenga.

Asegurase de que se está construyendo el producto correcto y de manera correcta es parte ineludible de entregar un producto con calidad. Para esto existen varias pruebas, técnicas y herramientas que podrán ser utilizadas a lo largo del desarrollo de software.

Tener una base cuantitativa a lo largo del proyecto es una forma de comprenderlo mejor. La mejor manera de lograrlo es utilizando métricas a lo largo del proceso de desarrollo.

Justificación

Las preguntas plateadas al inicio del proyecto se presentan de manera constante actualmente puesto que existen varias propuestas en el mercado y algunas de ellas con la promesa de lograrlo. Pero el hecho de tratar de responderla requiere más que sólo conocimiento técnico. Recordemos que este último sólo es el medio de apoyo que viene después del entendimiento e interpretación de las condiciones dadas. Existe una amplia bibliografía al respecto y se ha dicho bastante acerca del tema, pero aún con todo ello el software siguen dando de qué hablar y el tema no termina por definirse en su totalidad, por lo que es lícito para todos aquellos interesados el adentrarse y proporcionar nuevas propuestas que clarifiquen, enriquezcan o identifiquen nuevos rumbos. Todo con el fin de que las organizaciones puedan alcanzar los objetivos trazados u otros que sean de importancia.

Objetivo

Elaborar una propuesta para mejorar el desarrollo de software en la jefatura de una empresa privada.

Los objetivos específicos son:

- Atacar los problemas más importantes para la empresa.
    - Ordenar el desarrollo de software.
    - Asegurar la calidad del desarrollo de software.
    - Proporcionar una herramienta cuantitativa para los proyectos de desarrollo de software.
- Utilizar en la propuesta las mejores prácticas existentes en la industria del software.

Limitaciones

Como este proyecto forma parte de un plan de calidad, los objetivos que se pretenden alcanzar responden por su prioridad a puntos que tienen una vinculación directa con la estrategia de la organización. De la lista de problemas proporcionados anteriormente algunos se han traducido, conjuntado y acordado en tres puntos, que son los que se encuentran contenidos en el objetivo de este proyecto. Los demás problemas que integran la lista son postergados a otra fase del plan de calidad.

El proyecto abarca la investigación y diseño de la propuesta para la implantación de mejores prácticas en el ámbito particular de la empresa analizada. La puesta en marcha y evaluaciones posteriores no son contempladas en este proyecto.

Debido a que existe una amplia variedad de opciones para alcanzar los objetivos del proyecto. No se pretende por lo tanto abarcar todas estas opciones, sino presentar y aplicar aquellas a las que se tenga acceso y ajusten a las características del proyecto, por lo que no pretende ser un tipo de receta de cocina para todos los casos aun por muy similar que sea.

Capitulo II Marco teórico

Estas son algunas abreviaturas que se utilizarán en el documento:

Software Engineering Institute (SEI)
Software Engineering Body of Knowledge (SWEBOK)
Project Management Body of Knowledge (PMBOK)
Project Management Institute (PMI)
Institute of Electrical and Electronics Engineers (IEEE)
Capability Maturity Model Integration (CMMI)
Personal Software Process (PSP)
Team Software Process (TSP)
International Organization for Standardization (ISO)
Total Quality Management (TQM)
International Electrotechnical Commission (IEC)
Software Process Improvement and Capability dEtermination (SPICE)
Joint Technical Committee (JTC)
North Atlantic Treaty Organization (NATO)
Electronic Industries Alliance (EIA)
Software Engineering Laboratory (SEL)
National Aeronautics and Space Administration (NASA)

Modelo de Procesos para la Industria de Software (MoProSoft)
Evaluación de Procesos para la Industria de Software (EvalProSoft)
Secretaría de Economía (SE)
Plan Nacional de Desarrollo (PND)
Programa para el Desarrollo de la Industria de Software (PROSOFT)
Normalización y Certificación Electrónica (NYCE)
Área Metropolitana de Monterrey (AMM)
Zona Metropolitana del Distrito Federal (ZMDF)
Instituto Nacional de Estadística, Geografía e Informática (INEGI)
Producto Interno Bruto Nacional (PIBN)
Tecnologías de la Información (TI)
Micro, Pequeña y Mediana empresas (MPyMEs)

En 1968, en la conferencia de NATO en Garmish (Alemania), se manifestó la precaria situación en la que se encontraba el desarrollo de software a comparación de otras disciplinas ante la demanda de nuevos sistemas. Es por ello que se acuña el término “crisis del software” (*crisis gap*) para señalar la gran cantidad de fallas, exceso de costos y tiempo de entrega. Desde ese momento hasta nuestros días se ha escrito mucho al respecto con tal de solucionar este problema. Para este proyecto será de gran importancia recolectar parte de esta experiencia generada para tratar de subsanar los problemas planteados, ya que muchos de los temas tienen relación con el proyecto.

El software y sus características

Existen cualidades primarias que forman parte de la existencia del objeto y cualquier ausencia de estas cualidades hace imposible la existencia del objeto. A estas cualidades es a las que se refiere la siguiente definición de software:

El software es un conjunto de algoritmos ejecutados en una computadora. Un algoritmo es una secuencia de instrucciones lógicas y matemáticas con un objetivo en específico.

Intentar definir un objeto resulta difícil cuando ésta no se mantiene constante y en este caso el software no es la excepción. Ser un producto completamente confeccionado por el esfuerzo del hombre le otorga en cierta forma la posibilidad de ser modificado según sea la preferencia, pues entiende las relaciones constitutivas y puede moverse bajo este marco definido por él mismo. El hombre se mantiene en constante movimiento, por lo que no es raro que los objetos alcanzados por él no se mantengan estáticos por mucho tiempo. Este movimiento al que está sometido el hombre no sólo aplica a las cosas que le circundan, sino que también lo atrapan y al transformar también puede transformarse. Esto último lo podemos ver claramente de la siguiente manera: "El hombre expresa su ser y lo transforma al expresarlo. En cada momento es capas de ofrecer alguna particularidad, que siendo inesperada, es al mismo tiempo congruente con su ser."[1] Esto quiere decir que las definiciones nos permiten introducir en el tema propuesto desde algún punto de vista, por lo que no sería recomendable optar por una postura dogmática y considerar a una como privilegiada por sobre las demás, pues estas cambian o se generan constantemente. Y la definición proporcionada de software parte desde la perspectiva en la se origina el software y se presenta de manera comprensible para las pretensiones de este proyecto.

Las características primarias que le dan existencia al software son conocimientos sin propósito alguno. El conocimiento no es bueno ni malo, sólo es verdadero. Podría decirse que en este estado no son susceptibles de valoración (algo que sin duda podría discutirse). Pero una vez que se colocan fines al software es cuando son susceptibles a ser valorados y se convierten en un bien. Un bien es una cosa más el valor que se le ha incorporado.

Algunas características del software nos permitirán identificarla de otras cosas y que son posteriores a de las características primarias se encuentran:

- No tiene existencia física.
- Es una herramienta, un medio.
- La existencia del software sólo se da con la ayuda del hardware o computadora.
- Su habitad cambia constantemente.
- Depende de los requerimientos.
- Por lo general se construye a la medida.
- El software no se fabrica: se crea, se construye.
- El desarrollo del software propiamente dicho es un trabajo intelectual.
- Algunas partes se pueden utilizar en varios proyectos.
- El requerimiento puede surgir de un deseo o una necesidad.
- El software no se estropea, ni desgasta sólo se hace obsoleto.
- Es software es un producto del hombre.
- A medida que crece se dificulta su control.
- El producto de software se puede modificar.
- Etc.

[1] Nicol, Eduardo. *La idea del hombre*. México, FCE, 2003, p. 11.

Desde la perspectiva que tienen los modelos de calidad, el software debe de tener ciertas características que permitan determinar su calidad, de entre las más comunes podemos encontrar las siguientes:

- Corrección
- Confiabilidad
- Robustez
- Desempeño (*performance*)
- Usable
- Verificable
- Mantenible
- Reparable
- Evolucionable
- Reusable
- Portable
- Entendible
- Interoperable

Estas últimas características son las que permitirán evaluar al software y algunas de ellas (dependiendo del modelo) se llegan a dividir en subcaracterísticas y atributos. Los cuales pueden interactuar unos con otros.

Los requerimientos establecen sentido y la dirección que deberán de tomar los elementos del software (algoritmos ejecutados en una computadora). Las características de calidad pueden o no estar presentes sin que se vea afectada la existencia del software, puesto que evalúan al software una vez que éste ya existe y bajo una perspectiva específica.

Podremos decir que las características primarias son aquellas que posibilitan la existencia del software. Las características secundarias son derivadas de las primarias y nos permiten diferenciar una cosa de otra. Y las características de tercer nivel nos permiten valorar a las características primarias una vez que se le definen los fines por medio de los requerimientos. Se entiende en este documento por funciones a las características primarias que tienen fines determinados.

Las características de tercer nivel se han añadido conforme pasa el tiempo, esto no quiere decir que sean todas o deban de utilizarse por completo cada vez que se desarrolla el software. En cada desarrollo se encuentran diferentes intereses, lo que colocará a algunas características de tercer nivel sobre otras con el fin de que sean valoradas y quizá a otras no será necesario valorarlas.

Cada desarrollo tiene sus propias funciones y características de tercer nivel para valorarlas. Identificarlas es una habilidad que debe ser desarrollada por quienes se encargan de desarrollar el software. Sin olvidar que en la construcción de las funciones y valoración se involucran factores como: las personas, el tiempo de desarrollo, los recursos, la tecnología, etc. Lograr la correspondencia entre lo realizado y lo solicitado a través de las características de tercer nivel hace posible encontrar la distancia que hay entre las dos.

La calidad en el software

A la calidad también resulta difícil designarle una definición final. Pero con todo ello es posible reconocerla de manera intuitiva, pero sin llegar a un acuerdo general de lo que eso representa.

Se han generado varios intentos por definir la calidad. A continuación se presentan sólo algunos de estos intentos con el propósito de contar con un punto de partida.

- La capacidad de cumplir con las necesidades y expectativas del cliente (Feigenbaum).
- La mínima pérdida de un producto o servicio ocasionada a la sociedad desde que es entregado (Taguchy).
- Adecuación al uso (Juran).
- Conformidad con los requerimientos del cliente (Crosby).
- Grado predecible de uniformidad y fiabilidad a bajo costo y adecuado a las necesidades del mercado (Deming).
- Diseñar, producir y ofrecer un bien o servicio que sea útil, lo más económico posible y siempre satisfactorio para el cliente (Ishikawa).
- Grado en el cual un conjunto de características inherentes satisface requerimientos (ISO 9001).

Para David A. Garvin, la calidad se da por medio de diferentes puntos. Es por esto que las describe desde las siguientes cinco aproximaciones:

- Visión trascendental. La calidad se puede reconocer pero no definir. La calidad es un ideal hacia el que nos dirigimos pero no es posible alcanzarlo por completo.
- Visión del usuario. La calidad es adecuación al propósito.
- Visión de manufactura. La calidad es conformidad con la especificación. Considera a la calidad durante la producción y después de la distribución.
- Visión del producto. La calidad está vinculada con las características inherentes del producto. Se asoma al interior y evalúa las características inherentes del producto.
- Visión basada en valor. La calidad depende de la cantidad de dinero que el usuario está dispuesto a pagar por el producto. Iguala la calidad con lo que el cliente está dispuesto a pagar, un balance entre costo y calidad.

Esta clasificación permite englobar y ordenar a las definiciones existentes de calidad. También presenta ocho dimensiones de la calidad:

- Presentaciones. Son las características funcionales primarias del producto.
- Peculiaridades. Son complementos extra de las funciones primarias.
- Confiabilidad. Es la probabilidad de que un producto deje de funcionar adecuadamente dentro de un lapso determinado de tiempo.
- Conformidad con las especificaciones. Grado en el que el diseño y las características funcionales de un producto cumplen con las normas establecidas.
- Durabilidad. Lapso de vida de un producto antes de que sea necesario sustituirlo.
- Disponibilidad. Es la rapidez, cortesía, efectividad y facilidad de reparación.
- Estética. Son los aspectos sensibles que reflejan las preferencias de un individuo.
- Calidad percibida. La presentación del producto que se le hace al usuario.

Para abordar la calidad de manera general requiere de un gran esfuerzo y para acotar el universo de estudio y apegarnos a los fines del proyecto bastará con la pregunta ¿qué es el software de calidad? Para responder a esta pregunta parece ineludible el tema del valor, pues ya sea que con la calidad podemos determinar el valor del software o que con la valoración podemos determinar la calidad del software. Esto nos lleva a otra pregunta ¿qué son los valores? Este, es un tema que implica varios retos abordados como objeto de estudio por la axiología.[2] Entorno al valor se han dicho varias cosas sobre él como: que son inherente a las cosas, esencias, estados psicológicos,

[2] Disciplina que estudia los valores

entes parasitarios porque necesitan de un depositario, etc. Para el caso del software se podrían derivar las siguientes preguntas: ¿de qué manera se le agrega valor al software?, ¿cómo puede reconocérsele un valor al software?, ¿qué o quién decide si tiene o no calidad el software?, ¿un software valioso es igual a un software con calidad?, ¿de principio el software cuenta con un valor que le acompaña o todos le son agregados?, ¿cuáles son los tipos de valor que pueden aplicársele al software y bajo qué contexto se justifican?, etc. Estas son preguntas importantes para la calidad del software, pero por el momento bastará con los elementos que se cuenta actualmente.

A continuación se presentan unos intentos por definir la calidad del software:

- Concordancia del software producido con los requisitos explícitamente establecidos, con los estándares de desarrollo expresamente fijados y con los requisitos implícitos no establecidos formalmente, que desea el usuario (Pressman).
- Calidad del software es el grado en el que un sistema, componente o proceso cumple los requisitos especificados y las necesidades o expectativas del cliente o usuario (IEEE).

Basadas en las anteriores definiciones se han elaborado varias propuestas que representan la perspectiva que defienden. Por mencionar un ejemplo de ellos diremos que existen modelos[3] que mencionan las características o atributos que deben considerarse en la calidad de los productos de software y que de ellas pueden derivarse métricas, pruebas, procesos, técnicas, estándares, formatos y hasta software de apoyo. Cada uno de estos elementos pueden encontrarse en varias propuestas, por lo que algunas veces no es necesario volverlos a elaborar y simplemente son colocado de acuerdo con el planteamiento de la propuesta. Podríamos decir que existen varias perspectivas desde las cuales se puede ver y seleccionar un camino hacia la calidad del software. Cada perspectiva cuenta con un grupo de herramientas que la apoyan y características de interés que le otorgaran un valor diferente a comparación de otras. Las opciones representan sólo una posibilidad para solucionar un problema relacionado con la calidad del software y no la solución en sí.

Procesos

Los procesos reúnen a tres de las perspectivas desde las que una organización puede mejorar:

- Procedimientos y métodos
- Gente
- Herramientas y equipo

Mejorar la calidad por medio de procesos es una postura ha sido adoptada desde hace tiempo por la industria manufacturera y de servicios, pero en el desarrollo de software esto apenas está ocurriendo. Entre las ventajas mencionadas por quienes los han utilizado se encuentra el apoyo a la organización para alcanzar sus objetivos debido a que los ayuda a trabajar de manera inteligente y con una mejora constante. Otras ventajas que se postulan son: permite alinear la forma en la que se hacen negocios, es escalable y permite incorporar conocimiento de cómo hacer mejor las cosas, se puede examinar los recursos y tendencias, provee una infraestructura que puede tratar con el cambio y maximizar el personal y la tecnología para que sean más competitivas.

En la década de 1930, Walter Sheward inició su trabajo en la mejora de procesos con sus principios de control estadístico de la calidad, principios que fueron refinados por W. Edwards Deming y Joseph Juran. Más adelante Watts Humphrey, Ron Radice y otros, extendieron y aplicaron estos principios al software en IBM y en el SEI. Dando como resultado que este último haya tomado como principio el siguiente:

[3] Pueden consultar los modelos de McCall, ISO/IEC 9126, Boehm.

"La calidad de un sistema o un producto es altamente influenciada por la calidad del proceso utilizado para desarrollarlo o mantenerlo"[4]

De entre los prejuicios más comunes hacia los procesos se pueden encontrar los siguientes:

- Interfiere con la creatividad.
- Es igual a burocracia.
- No son necesarios cuando se construyen prototipos.
- Es sólo útil en proyectos largos.
- Interfiere en la agilidad dentro de los mercados que se mueven constantemente.
- Son costos.
- No son necesarios cuando se tiene un buen equipo de trabajo, tecnología de punta y un administrador experimentado.

Las ventajas que prometen los procesos se han convertido en motivo para que las organizaciones busquen mejorar sus procesos, dando como resultado la generación de varios modelos y estándares de referencia (Fig. **1**) Pero en este caso sólo se abordará brevemente aquellos que cuentan con mayor difusión a nivel mundial[5] (ISO 9001, CMM e ISO/IEC 15504 [SPICE]) y uno más que es de reciente creación (MoProSoft).

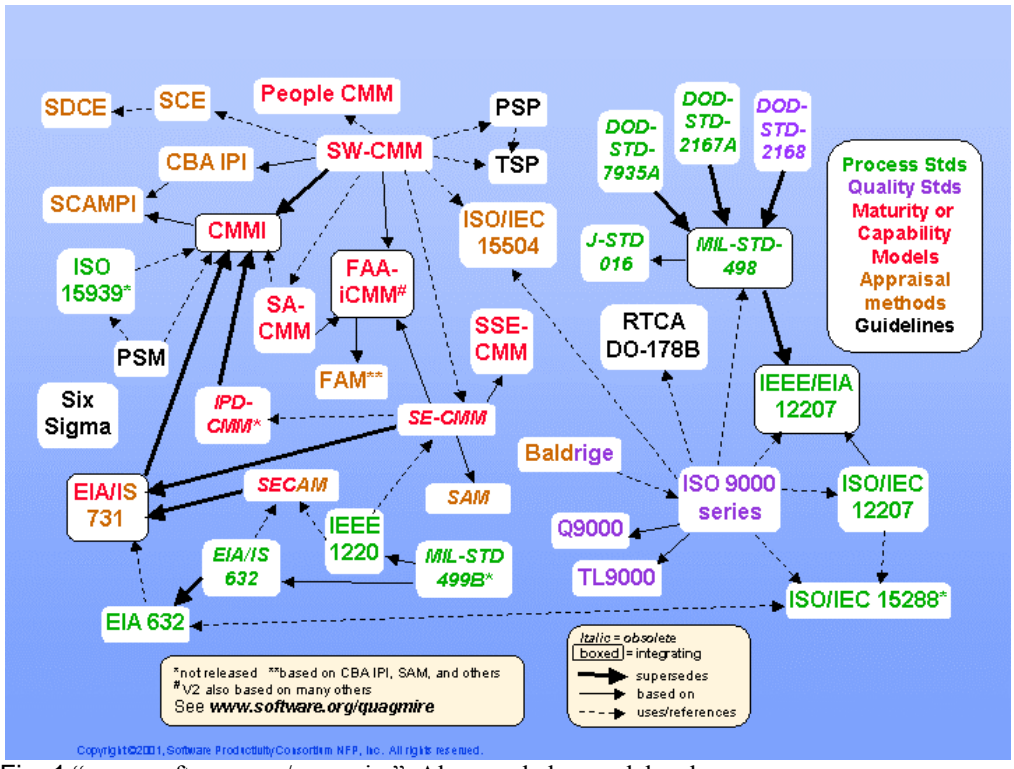


Fig. 1 "www.software.org/quagmire". Algunos de los modelos de proceso.

[4] Chrissis, Mary Beth, et. al. *CMMI Guidelines for process integration and product improvement*. México, Addison-Wesley, 2003. p. 5.

[5] Wang Y., et. al. "Towards Software Process Excellence: A survey report on the best practices in the software industry" en *ASQ Journal of Software Quality Professional*, Vol. 2, No. 1, pp.34-43 (1999)

ISO 9001

Las normas de la serie ISO 9000, son un conjunto de normas, utilizadas como marco para diseñar, implantar y certificar sistemas de gestión de calidad. El término ISO no proviene de un acrónimo, sino de la palabra griega que significa “igual”.

La ISO es una organización mundial no gubernamental, compuesta por representantes de los organismos de normalización nacional, con sede en Ginebra. Dicha organización se articula en comités técnicos que se encargan de la elaboración de las normas internacionales, y dichos comités están integrados por miembros de los organismos federados interesados en el objeto de trabajo de la comisión. Una vez elaborado el proyecto de norma, éste es enviado a los organismos miembros para su aprobación, la cual requiere el voto favorable de al menos dos terceras partes de los organismos miembros del comité. Tras su aprobación las normas son difundidas internacionalmente a través de los organismos nacionales federados.

Los estándares se aplican para demostrar que se cuenta con un nivel de experiencia en el diseño y construcción de un producto. Son usadas para regular la calidad interna y asegurar la calidad de los proveedores.

La familia de normas ISO 9000 fue publicado por primera vez en 1987, revisado en 1994 y actualizado nuevamente en el año 2000 (con un compromiso de ser revisado cada 5 años).

Dentro de la familia de estándares del ISO 9000 podemos encontrar:

- ISO 9000, Fundamentos y vocabulario.
- ISO 9001, Requisitos para aseguramiento de la calidad.
- ISO 9004, Directrices para la mejora del rendimiento.
- ISO 9011, Directrices para la auditoría de los sistemas de gestión de la calidad y/o ambiental.

De los estándares del ISO 9000, el ISO 9001 por ser aplicado a las actividades relacionadas con el software. La presentación del ISO 9001:2000 es general, pero existe una interpretación se encuentra en el ISO 9000-3, que es una guía para las organizaciones en la aplicación del ISO 9001:2000 para adquisición, abastecimiento, desarrollo, operación y mantenimiento de software y servicios de soporte relacionados.

La norma ISO 9001:2000 puede utilizarse para cualquier tipo de industria por su carácter “genérico”. La norma especifica requerimientos generales que se deben cumplir por elementos y procesos concretos. Es un enfoque de gestión de calidad basada en procesos. Este enfoque facilita la ejecución y gestión de actividades, así como ciclos de mejora continua siguiendo el principio planificar-ejecutar-verificar-actuar (PDCA por sus siglas en inglés). El flujo de la información generada por las mediciones son proporcionadas a los responsables de tomar decisiones y corregir planes. ISO 9001 se centra en la eficacia del sistema de gestión de la calidad para dar cumplimiento a los requisitos del cliente.

ISO 9001: 2000 “Sistema de gestión de calidad – requisitos” está estructurado en ocho secciones:

1. Objetivo y campo de aplicación
   1.1. Generalidades
   1.2. Aplicación
2. Referencias normativas
3. Términos y definiciones
4. Sistema de gestión de calidad
   4.1. Requisitos generales
   4.2. Requisitos de documentación

- 4.2.1.Generalidades
- 4.2.2.Manual de calidad
- 4.2.3.Control de registros de la calidad

5. Responsabilidad de la dirección
   - 5.1. Compromiso de la dirección
   - 5.2. Enfoque al cliente
   - 5.3. Política de calidad
   - 5.4. Planificación
     - 5.4.1.Objetivos de la calidad
     - 5.4.2.Planificación del sistema de gestión de calidad
   - 5.5. Responsabilidad, autoridad y comunicación
     - 5.5.1.Responsabilidad y autoridad
     - 5.5.2.Representante de la dirección
     - 5.5.3.Comunicación interna
   - 5.6. Revisión por la dirección
     - 5.6.1.Generalidades
     - 5.6.2.Información para la revisión
     - 5.6.3.Resultados de la revisión
6. Gestión de recursos
   - 6.1. Provisión de recursos
   - 6.2. Recursos humanos
     - 6.2.1.Generalidades
     - 6.2.2.Competencia, toma de conciencia y formación
   - 6.3. Infraestructura
   - 6.4. Ambiente de trabajo
7. Realización del producto
   - 7.1. Planificación de la realización del producto
   - 7.2. Procesos relacionados con el cliente
     - 7.2.1.Determinación de los requisitos relacionados con el producto
     - 7.2.2.Revisión de los requisitos relacionados con el producto
     - 7.2.3.Comunicación con el cliente
   - 7.3. Diseño y desarrollo
     - 7.3.1.Planificación del diseño y desarrollo
     - 7.3.2.Elementos de entrada para el diseño y desarrollo
     - 7.3.3.Resultados del diseño y desarrollo
     - 7.3.4.Revisión del diseño y desarrollo
     - 7.3.5.Verificación del diseño y desarrollo
     - 7.3.6.Validación del diseño y desarrollo
     - 7.3.7.Control de cambios del diseño y desarrollo
   - 7.4. Compras
     - 7.4.1.Proceso de compras
     - 7.4.2.Información de las compras
     - 7.4.3.Verificación de los productos comprados
   - 7.5. Producción y presentación del servicio
     - 7.5.1.Control de la producción y presentación del servicio
     - 7.5.2.Validación de los procesos
     - 7.5.3.Identificación y trazabilidad
     - 7.5.4.Propiedad del cliente
     - 7.5.5.Preservación del producto
   - 7.6. Control de los dispositivos de seguimiento y medición
8. Medición, análisis y mejora
   - 8.1. Generalidades
   - 8.2. Seguimiento y medición
     - 8.2.1.Satisfacción del cliente
     - 8.2.2.Auditoría interna
     - 8.2.3.Seguimiento y medición de los procesos

8.2.4.Seguimiento y medición del producto
8.3. Control del producto no conforme
8.4. Análisis de datos
8.5. Mejora
8.5.1.Mejora continua
8.5.2.Acción correctiva
8.5.3.Acción preventiva

CMMI

En 1984, el departamento de defensa de los Estados Unidos de América establece al SEI de la Universidad de Carnegie Mellon, para ayudarlo en la valoración de sus contratistas. CMM representó para el departamento de defensa que los productos y servicios fueran los más asequibles para sus armas de guerra.

Mark Paulk y otros en el SEI crearon el primer modelo de madurez de capacidad, diseñado para organizaciones de desarrollo software. SW CMM había sido el principal producto del SEI, liberado en 1991. Este modelo permitió a las organizaciones a mejorar la eficiencia en el desarrollo de la calidad de los productos de software.

El modelo de madurez de las capacidades es un modelo de referencia de prácticas maduras en una disciplina específica, utilizada para evaluar la capacidad de los grupos para desempeñar esa disciplina.

CMM dirige su enfoque a la mejora de procesos en una organización, estudia los procesos de desarrollo y produce una evaluación de la madurez de la organización según una escala de cinco niveles: inicial, repetible, definido, dirigido y optimizado. Los modelos contienen los elementos esenciales de procesos efectivos para una o más disciplinas y describen el camino para evolucionar y mejorar desde procesos inmaduros a procesos disciplinados, maduros con calidad y eficiencia mejorada y probada.

CMM es un enfoque general para las organizaciones De manera que se desarrollaron formas de abordarlo de manera más específica, es por ello que surgieron TSP y PSP. Aunque no son obligatorios para CMM, resulta importante mencionarlos por su aportación.

Watts Humphrey, aplica los principios subyacentes de CMM a las prácticas de desarrollo individual. PSP es diseñado para ser un CMM nivel cinco para desarrollo personal.

PSP enseña a los ingenieros a:

- Administrar la calidad de sus proyectos.
- Hacer compromisos que ellos pueden cumplir.
- Mejorar estimaciones y planificaciones.
- Reducir los defectos en sus productos.

El personal constituye gran parte de los costos del desarrollo de software, las habilidades de los ingenieros ampliamente determinan los resultados del proceso de desarrollo de software. PSP puede ser usado por ingenieros como guía o como un disciplinado y estructurado acercamiento al desarrollo de software. PSP es un prerrequisito para una organización que pretende introducir TSP.

PSP puede ser incluida en muchas partes del proceso de desarrollo de software, incluyendo:

- Desarrollo de pequeños programas.
- Definición de requerimientos.

- Elaboración de documentos.
- Pruebas de sistemas.
- Mantenimiento de sistemas.
- Mejora de grandes sistemas de software.

Aunque las prácticas de PSP eran posibles y daban resultados, era casi imposible mantener la disciplina de las prácticas si el ambiente circundante no los animaba y exigía. Es por esto que Humphrey desarrolló TSP para unidades pequeñas de desarrollo (las cuales son comunes en las organizaciones). TSP fue diseñado para ser un CMM nivel cinco para equipos.

TSP junto a PSP ayudan al ingeniero a:

- Asegurar la calidad de los productos de software.
- Crear productos de software confiables.
- Mejorar le proceso de administración en una organización.

Usando TSP, una organización puede construir equipos autodirigidos que planifican y siguen su trabajo, estableciendo metas y sus propios procesos y planes de trabajo. TSP ayuda a las organizaciones a madurar y disciplinar prácticas de ingeniería que producen seguridad.

Debido al éxito de SW CMM y a la demanda de modelos en otras áreas, el SEI desarrolló otros modelos de CMM, conformados de la siguiente manera:

- *Systems engineering* SE – CMM
- *Integrated product development* IPD – CMM
- *Software Acquisition* SA – CMM
- *Human resources* People – CMM
- *Software* SW - CMM

Aunque los modelos habían sido útiles para muchas organizaciones, el uso de múltiples modelos se había vuelto problemático. Muchas organizaciones querían enfocar sus esfuerzos a través de diferentes disciplinas. Pero la diferencia entre cada uno de los modelos, arquitectura, contenido y acercamiento, limitaban la habilidad de las organizaciones para enfocar sus mejoras de manera exitosa. Aplicar múltiples modelos que no se encontraban integrados se volvía costoso en términos de capacitación, valoración y actividades de mejora.

CMMI fue elaborado para solucionar el problema de múltiples CMMs. El objetivo consistía en combinar tres modelos:

- SW – CMM
- System engineering capability model (EIA 731)
- IPD - CMM

Estos tres modelos fueron seleccionados por su amplia aceptación y los diferentes accesos para mejorar procesos en una organización. CMMI fue liberado por el SEI en 2002.

CMMI presenta dos representaciones del modelo: continua o escalonada. La representación continua tiene una sola área de proceso o un conjunto de áreas de proceso (Fig. 2) y la representación escalonada tiene un conjunto establecido de áreas de proceso a lo largo de la organización con cinco niveles (Fig. 3).

Las áreas de procesos son un conjunto de prácticas relacionadas para llevar a cabo un conjunto de metas. Son el elemento principal para establecer la capacidad de los procesos de una organización.

Niveles de capacidad por representación continua:

- Nivel 5 Optimizado
- Nivel 4 Administrado cuantitativamente

- Nivel 3 Definido
- Nivel 2 Administrado
- Nivel 1 Desempeñado
- Nivel 0 Incompleto

Niveles de madurez por representación escalonada:

- Nivel 5 Optimizado
- Nivel 4 Administrado cuantitativamente
- Nivel 3 Definido
- Nivel 2 Administrado
- Nivel 1 Desempeñado

Categoría	Área de proceso
Administración de proyectos	Planificación del proyecto Monitoreo y control del proyecto Administración del acuerdo con el proveedor Administración integrada del proyecto Administración de riesgos Administración cuantitativa de proyectos
Soporte	Análisis causal y resolución Análisis de decisión y resolución Medición y análisis Aseguramiento de la calidad del producto y proceso Administración de la configuración
Ingeniería	Administración de requerimientos Desarrollo de requerimientos Solución técnica Integración del producto Verificación Validación
Administración de procesos	Enfoque en procesos organizacionales Definición de procesos organizacionales Capacitación organizacional Desempeño de procesos organizacionales Innovación y despliegue organizacional

Fig. 2 Áreas de procesos para una organización continua.

Nivel	Centrado en	Áreas de proceso
Optimizado	Mejora continua de procesos	Innovación y despliegue organizacional Análisis causal y resolución
Administrado cuantitativamente	Administración cuantitativa	Desempeño de procesos organizacionales Administración cuantitativa de proyectos
Definido	Estandarización de procesos	Desarrollo de requerimientos Solución técnica Integración del producto Verificación Validación Enfoque en procesos organizacionales Definición de procesos organizacionales Capacitación organizacional Administración integrada del proyecto Administración integrada de proveedores Administración de riesgos Análisis de decisión y resolución

Nivel	Centrado en	Áreas de proceso
		Medio ambiente organizacional para la integración Integración de equipos
Administrado	Administración básica de proyectos	Administración de requerimientos Planificación del proyecto Monitoreo y control del proyecto Administración del acuerdo con el proveedor Medición y análisis Aseguramiento de calidad de producto y proceso Administración de la configuración
Desempeñado		

Fig. 3 Áreas de procesos por nivel de madurez

ISO/IEC 15504 (SPICE)

Es un estándar internacional de evaluación y determinación de la capacidad y mejora continua de procesos para la ingeniería de software. Es aplicable a cualquier organización que quiera mejorar la capacidad de cualquiera de sus procesos de software. Se puede utilizar como herramienta de evaluación del estado de los procesos de software de la empresa. Es independiente de la organización, modelo del ciclo de vida, metodología y tecnología.

En 1991 se aprueba la investigación para elaborar un estándar de evaluación de procesos de software, lográndose un consenso internacional en 1992. En 1998 se publica la primera versión del estándar como informe técnico.

La evolución hacia un estándar internacional se ha constituido de tres fases: la fase uno en 1995, la fase dos de 1996 a 1998 y la fase tres hasta marzo del 2003. Posteriormente comienza la fase de *benchmarking* para recolectar datos de los procesos de evaluación y analizarlos y comienza la publicación de partes del estándar.

ISO/IEC 15504 se apoya de la escala de puntuación de capacidad de CMM, las actividades de proceso de ingeniería de ISO/IEC 12207, Trillium y CMM, la representación de capacidad basada en perfiles de atributos de BOOTSTRAP y la experiencia del sistema de gestión de la calidad general de ISO 9001.

Se desarrolla un modelo de evaluación de la capacidad del proceso, donde se valora la organización de desarrollo software en la dimensión del proceso contra los atributos del proceso en la dimensión de capacidad. La estructura del modelo se encuentra dividida en cinco partes:

- Parte 1. Conceptos y Vocabulario. En publicación (7/10/2004)
- Parte 2. Realizando una Evaluación (Requisitos, normativa). Publicado (15/10/2003).
- Parte 3. Guía para Realización de Evaluaciones. Publicada (15/01/2004)
- Parte 4. Guía para el Uso de Resultados de Evaluaciones. Publicada (01/07/2004)
- Parte 5. Un Modelo de Evaluación de Procesos Ejemplar (01/03/2006)

Inicialmente se contaba con treinta y cinco procesos agrupados en cinco categorías. Sin embargo, Con el propósito de expandir su aplicación del estándar a más allá de un determinado ciclo de vida, se permite la evaluación del estándar para aceptar modelos de referencia de procesos y eliminando la inicial dimensión de procesos. La medida de capacidad es aplicable a cualquier modelo de procesos plasmado en un modelo de referencia de procesos compatible con ISO 12207.

ISO/IEC 15504 – 2 establece los requisitos para modelos de procesos de referencia y para los métodos de evaluación sin establecer uno en particular. Define el marco de medición de capacidad

de proceso[6]. Este marco cuenta con una escala de seis niveles y nueve atributos[7] (los atributos tienen una escala de cuatro valores) (Fig. 4).

Nivel de capacidad	Atributos	Descripción del nivel
0 Incompleto	Ninguno	Generalmente no se obtienen los propósitos del proceso. No se identifican fácilmente cuales son los productos del trabajo o salidas del proceso.
1 Realizado	1.1 Realización del proceso	Generalmente se alcanza el propósito del proceso. Aunque puede no estar planificado rigurosamente ni rastreado. Están identificados los productos del proceso que testifican que se alcanzó el objetivo.
2 Gestionado	2.1 Gestión de la ejecución 2.2 Gestión de productos	El proceso genera productos de una calidad aceptable en tiempos razonables. Los productos siguen los estándares establecidos. Se siguen procedimientos especificados, planificados y revisables.
3 Establecido	3.1 Definición del proceso 3.2 Recursos del proceso	El proceso se efectúa siguiendo buenos principios de ingeniería de software. Cada implementación individual del proceso sigue estándares aprobados, revisados y documentados.
4 Predecible	4.1 Medida del proceso 4.2 Control del proceso	El proceso está definido y se sigue constantemente en la práctica con límites controlados para alcanzar sus objetivos. La ejecución se administra objetivamente. La calidad de los productos de trabajo se conoce cuantitativamente. Se toman mediciones detalladas y se analizan. Esta práctica lleva a que se entienda cuantitativamente la capacidad del proceso y se mejore la posibilidad de predecir la ejecución.
5 Optimizado	5.1 Cambio de proceso 5.2 Mejora continua	La ejecución del proceso se optimiza para cubrir las necesidades actuales y futuras y los procesos alcanzan repetidamente sus objetivos. El proceso se monitorea constantemente contra sus objetivos para retroalimentarse cuantitativamente y mejorar analizando los resultados. La innovación del proceso se alcanza probando ideas y tecnologías innovadoras y cambiando lo que no sea efectivo. Se establece la cuantificación de la efectividad y eficiencia de los procesos basados en las metas y los objetivos del negocio.

Fig. 4 Niveles y atributos del marco de medición ISO/IEC 15504

[6] La capacidad de proceso es el rango de resultados esperados que se obtienen siguiendo el proceso.
[7] Los atributos de un proceso se usan para determinar si el proceso ha alcanzado una cierta capacidad, midiendo un aspecto particular del proceso.

MoProSoft[8]

MoProSoft es un modelo de reciente creación, para la industria de software en México que fomenta la estandarización de su operación a través de la incorporación de las mejores prácticas en gestión e ingeniería de software.

Antecedentes

El PND 2001–2006 plantea el objetivo de elevar y extender la competitividad de México, una de las formas de lograrlo es mediante una estrategia que promueva el uso y aprovechamiento de la tecnología y de la información. Es por eso que la SE definió PROSOFT, como uno de los medios para concretar el objetivo. En 2002 la SE inició PROSOFT, que tiene como objetivo fortalecer la industria de software en México.

Los objetivos básicos de PROSOFT son:

- Alcanzar el promedio mundial de gasto en tecnologías de información.
- Lograr una producción de software de 5,000 millones de dólares anuales para el 2010.
- Convertir a México en el líder latinoamericano de soporte y desarrollo de servicios basados en TI.

PROSOFT está constituido por las siguientes estrategias:

1. Promover las exportaciones y la atracción de inversiones.
2. Educación y formación de personal competente en el desarrollo de software, en cantidad y calidad convenientes.
3. Contar con un marco legal promotor de la industria,
4. Desarrollar el mercado interno.
5. Fortalecer a la industria local.
6. Alcanzar niveles internacionales en capacidad de procesos.
7. Promover la construcción de infraestructura física y de telecomunicaciones.

La estrategia 6 está dividida en los siguientes rubros:
6.1 Formación de instituciones de capacitación y asesoría en mejora de procesos.
6.2 Definición de modelos de procesos y de evaluación apropiados para la industria de software mexicana.
6.3 Apoyo financiero para capacitación y certificación de la capacidad de procesos.
6.4 Premio Nacional de Calidad en Tecnologías de Información.
6.5 Estímulos fiscales al desarrollo tecnológico en las empresas.
6.6 Formación de un cajón de financiamiento para actividades de investigación y desarrollo.
6.7 Otros apoyos para actividades de investigación y desarrollo.

Referido al punto 6.2 es que se desprende la definición de un Modelo de Procesos para la Industria del Software (MoProSoft) y de evaluación apropiado para la industria del software mexicano (EvalProSoft).

[8] Información obtenida de: http://www.software.net.mx y http://www.amcis.org.mx

Situación de la industria del software en México

PROSOFT cuenta con varios estudios disponibles y de ellos se abordarán solamente dos, con énfasis en los puntos más importantes, con el fin de comprender las características en la que se encontraba la industria nacional de software al momento en que se realizaron dichos estudios y sobre las cuales se desenvuelven MoProSoft y EvalProSoft.

El "Estudio del perfil de la industria mexicana de software para definir los nichos de mercado internacional acordes al perfil y competitividad de la industria", en su fase uno y específicamente en el criterio dos "Perfil de la industria mexicana de software y servicios relacionados" se analizan las principales características de la oferta mexicana de software y servicios relacionados.

En este primer estudio encaminado a definir el perfil de la oferta nacional, destacan dos estudios realizados por las empresas BINARY y LEVANTA, así como el diseño y la aplicación de una encuesta realizada por la SE. También cuenta con datos complementarios como el censo económico del INEGI de 1998, IDC, OCDE y una encuesta de ESANE Consultores. Información que se presenta a continuación de manera resumida.

En 2003 el gasto total aproximado en productos de software fue de 800 millones de dólares, pero sólo una parte de ésta es satisfecha por productos nacionales, ya que cerca de un 90 por ciento del software empaquetado que se vende es importado. Mientras que se estima que el gasto en servicios de TI en el 2003 representó una cantidad de 1,965 millones de dólares, de los cuales gran parte es producida en el país, estimándose alrededor de un 90 por ciento.

Estimaciones indican que el número de empresas en la industria (incluyendo empresas de software y de servicios de TI) es de 1,500 y una planta laboral de 61,800 empleados. El perfil de la industria es mayoritariamente micro y pequeña (cerca del 83 por ciento son empresas con menos de 50 empleados).

El resultado de la encuesta a 70 empresas (principalmente desarrolladoras), indica que el 40 por ciento fueron creadas a partir del 2000 y más del 60 por ciento tienen menos de 10 años (al momento de realizarse la encuesta).

La oferta de la industria está enfocada hacia la provisión de servicios de TI. De las 73 empresas encuestadas: 84 por ciento genera servicios de desarrollo e integración, 81 por ciento servicios de consultoría y 70 por ciento servicios de mantenimiento y soporte de software La mayoría de sus productos y/o servicios los dirigen a cuatro sectores: manufactura, servicios financieros, gobierno y comunicaciones.

Sólo 27 de 70 empresas encuestadas destinan parte de sus ventas al extranjero, de los cuales EUA representa el primer destino de dichas exportaciones y en segundo lugar América Latina.

Aunque la mayoría de las empresas han adoptado algún modelo para mejorar la calidad y eficiencia de sus procesos, sólo optan por una certificación aquellos que la utilizan como una herramienta para exportar.

En México la certificación no constituye un requisito indispensable para que las empresas obtengan proyectos de desarrollo, y para las empresas que exportan tampoco deben acreditar una certificación con sus clientes. Por lo que pocas empresas enfrentan los altos costos de una certificación.

De la encuesta realizada, 15 empresas esperan acreditar un nivel de CMM para finales del 2004: 6 empresas en nivel 2, 8 empresas en nivel 3 y una empresa en nivel 5.

Por otra parte el "Estudio del nivel de madurez y capacidad de procesos de la industria de tecnologías de información"[9], pretende evaluar el estado en el que se encuentran los niveles de madurez y capacidad de procesos de las empresas de software y servicios relacionados en AMM y ZMDF.

De las empresas encuestadas con relación a la actividad principal que realizan[10]: el 39.8 por ciento se enfoca a servicio de programación de cómputo a la medida, el 35.9 por ciento a software empaquetado, el 20.3 por ciento a servicios de integración e implementación de sistemas y el 3.9 por ciento a servicios de administración y operación de TI.

En lo referente al tamaño de las empresas[11] el 92 por ciento son MPyMEs. Esto quiere decir que el 82 por ciento cuenta con menos de 51 empleados. Micros y pequeñas empresas representan el 79 por ciento de las empresas que declararon que su principal actividad eran los servicios de programación de cómputo a la medida. Las MPyMEs en su conjunto generan el 70 por ciento (siendo notoria la participación de las pequeñas empresas con un 36 por ciento) de los empleos directos y las empresas grandes generan el restante 30 por ciento.

De todo el personal el 24 por ciento son empleados de administración y ventas y el 76 por ciento es personal técnico[12]. Dentro de los empleados técnicos el 7 por ciento cuenta con una capacitación en mejora de procesos, ya sea ISO 9000, CMM, PSP/TSP, entre otros.

En cuanto a las exportaciones exclusivamente de software sólo 15 empresas realizan estas actividades, representando sólo el 12 por ciento del total de las empresas encuestadas.

El gasto anual promedio en actividades de calidad y mejora de procesos de desarrollo de software es de alrededor de 1.1 millones de pesos por empresa, destacando que las empresas grandes gastan el doble del promedio. Las MPyMEs invierten una mayor proporción de sus ventas en actividades de calidad[13].

Las empresas reportaron que la inversión en actividades de calidad y mejora de procesos de software medida contra los beneficios obtenidos es de alrededor de 1.3 en promedio.

Como resultado del nivel de madurez de cada unos de los procesos, se construyó un índice agregado de capacidad de procesos de las 100 empresas evaluadas[14]. El valor del índice agregado general fue de 0.9, lo que quiere decir que:

- En promedio, las empresas de software y servicios realizan estimaciones del trabajo y preparan agendas de actividades, el alcance del trabajo está definido y existe un enfoque

[9] Diagnóstico realizado por PROSOFT durante el último trimestre del año 2003 y el primer trimestre de 2004. Como resultado de los cuestionarios obtenidos de 128 empresas. Empresas localizadas en regiones en las que en su conjunto aportan el 30% de PIBN, según el censo económico del INEGI de 1999. Y cuya participación en el sector de las tecnologías de información cuentan con más del 50% de la inversión en equipo de cómputo, 40% de las unidades económicas de servicios de análisis de sistemas y procesamiento informático, 40% del personal ocupado en el mismo rubro y 60% del valor de la producción de servicios de análisis de sistemas y procesamiento informático.

[10] La clasificación en cuatro categorías de servicios es con base en el *North American Industry Classification System*.

[11] De acuerdo con la clasificación por número de empleados de la Ley para la Competitividad de la Micro, Pequeña y Mediana Empresas.

[12] Incluye gerentes, líderes de proyecto, analistas, ingenieros en software, personal de soporte y subcontratación de programadores.

[13] Son resultados obtenidos de las respuestas de 40 empresas, o sea, el 30% del total de las empresas bajo estudio; significa que aunque la información es válida, podría no ser representativa para la formulación de conclusiones generales.

[14] Basados en el modelo SPICE / ISO/IEC 15504.

estructurado para realizar el trabajo. Generalmente se logra el propósito del proceso, aunque no haya sido rigurosamente planeado ni seguido. Existen productos de trabajo identificables y existe un acuerdo generalizado de que las acciones deben ser ejecutadas cuando se requieren.

- En este nivel las personas realizan su trabajo de acuerdo a lo que consideran apropiado o relevante para el logro del propósito del proceso, que por lo general es ineficiente y poco productivo.

Con lo anterior, significa que el 66 por ciento de las empresas encuentran por debajo del índice agregado general, lo cual significa que sus procesos los realizan de manera "incompleta". Por lo que respecta al 34 por ciento de las empresas que se encuentra por encima del índice agregado general, se encuentran colocados de la siguiente manera: del nivel de madurez 0.9 a 1.9 el 15 por ciento de las empresas, del nivel 2 al 2.9 el 10 por ciento y del nivel 3 al 5 el 9 por ciento.

El porcentaje de MPyMEs que han logrado niveles superiores a 2 es superior al porcentaje de las empresas grandes, siendo la proporción de 4 a 1.

Con respecto a la metodología de desarrollo y calidad de software que utilizaron las empresas encuestadas los resultados son: 30 por ciento se apegaron a estándares internacionales (ISO 9000 o CMM/CMMI), el 28 por ciento aplica otro tipo de metodología no especificada y el 42 por ciento no utiliza ninguna metodología.

También se encontraron empresas que contaban con algún tipo de certificado o evaluación: 3 de las empresas encuestadas certificadas en ISO 9000, 7 en CMM/CMMI y 12 con otro tipo de certificación.

La norma

Características deseadas del modelo por parte de la industria eran:

- Específico para el desarrollo y mantenimiento de software.
- Fácil de entender (comprensible).
- Definido como un conjunto de procesos.
- Práctico y fácil de aplicar, sobre todo en organizaciones pequeñas.
- Orientado a mejorar los procesos para contribuir a los objetivos del negocio y no simplemente ser un marco de referencia de certificación.
- Debe de tener un mecanismo de evaluación o certificación, que indique un estado real de una organización durante un periodo de vigencia específico.
- Aplicable como norma mexicana.

Para lograr las características deseadas se analizaron los estándares disponibles.
SEI

- SW CMM 1993
- CMMI 2002

ISO

- ISO 9000:1994
- ISO/IEC 12207: 1995
- ISO/IEC TR 15504: 1998
- ISO/IEC 12207 Enmienda 1: 2002
- ISO/IEC 15504-2: 2003
- ISO 9000:2000

Del análisis de dichos modelos ninguno cumplió con las características deseadas, por lo que se decidió generar un modelo propio, acompañado de los elementos necesarios para su adecuada implantación y funcionamiento.

La norma contempla los siguientes elementos:

- Modelo de procesos (qué procesos)
- Modelo de capacidades de procesos (qué evaluar)
- Método de evaluación (cómo evaluar)

En NYCE se encuentra la norma mexicana NMX-059-NYCE-2005 bajo el nombre: Tecnología de la Información-Software-Modelos de procesos y de evaluación para desarrollo y mantenimiento de software. Compuesta con las siguientes partes:
Parte 01: Definición de conceptos y productos
Parte 02: Requisitos de procesos (MoProSoft)
Parte 03: Guía de implantación de procesos
Parte 04: Directrices para la evaluación (EvalProSoft)

La norma fue publicada en el Diario Oficial de la Federación el 15 de agosto del 2005.

Modelo de procesos

El modelo de procesos cuenta con tres categorías, seis procesos y tres subprocesos (Fig. 5).

Categoría	Proceso	Subproceso
Alta dirección	Gestión de negocios	Ninguno
Gestión	Gestión de procesos	Ninguno
	Gestión de proyectos	Ninguno
	Gestión de recursos	Recursos humanos y ambiente de trabajo
		Bienes, servicios e infraestructura
		Conocimiento de la organización
Operación	Administración de proyectos específicos	Ninguno
	Desarrollo y mantenimiento de software	Ninguno

Fig. 5 Estructura de MoProSoft

El modelo de procesos MoProSoft está dirigido a las empresas o **áreas internas** dedicadas al desarrollo y/o mantenimiento de software. Las organizaciones, que no cuenten con procesos establecidos, pueden usar el modelo ajustándolo de acuerdo a sus necesidades. Mientras que las organizaciones, que ya tienen procesos establecidos, pueden usarlo como punto de referencia para identificar los elementos que les hace falta cubrir.

El modelo se encuentra basado en las mejores prácticas internacionales con las siguientes características:

- Fácil de entender
- Fácil de aplicar
- No costoso en su adopción
- Ser la base para alcanzar evaluaciones exitosas con otros modelos o normas, tales como ISO 9000:2000 o CMM v 1.1

Método de evaluación

Cumple con los requisitos establecidos en ISO/IEC 15504 – 2

El método utiliza los requisitos de MoProSoft y el modelo de capacidades ISO/IEC 15504 – 2 para calificar los procesos. Nivel de madurez de capacidades de la organización es definido como el máximo nivel de capacidades alcanzado por todos los procesos.

EvalProSoft pretende:

- Evaluación del perfil de capacidades de procesos y de la madurez de capacidades de la organización
- Autoevaluación
- Evaluación por parte del comprador
- Efecto lateral: evaluación de la industria

Modelo de capacidades

Se encuentra basado en el ISO/IEC 15504 – 2 (Fig. 4)

SWEBOK

SWEBOK forma parte del esfuerzo por lograr que la ingeniería de software sea colocada como una disciplina legítima de la ingeniería y una profesión reconocida. Es por ello que desde 1993 la IEEE y la ACM, a través del SWECC, han promocionados ampliamente a la ingeniería de software como una profesión. Iniciando así el proyecto de SWEBOK en el año de 1998.

Es indudable que un cuerpo de conocimiento es fundamental para lograr que la ingeniería de software pueda evolucionar hacia profesionalización. El proyecto *Guide to the Software Engineering Body of Knowledge* es una iniciativa por lograr un consenso en el cuerpo de conocimiento. Proyecto en su versión del 2004 cuenta con una gran cantidad de observaciones y comentarios a nivel mundial que han hecho posible mejorarlo constantemente[15].

SWEBOK se describe como la suma de conocimiento dentro de la ingeniería de software. Como no es posible colocar en un documento toda la información de la disciplina, fue necesario colocada una guía de apoyo. La guía pretende identificar y describir un subconjunto del cuerpo de conocimiento que es “generalmente aceptado”.

Al decir “generalmente aceptado” se refiere a que éste sólo abordará en específico una sección de la clasificación que se ha realizado del conocimiento y que se presenta a continuación:

- Especializada. Prácticas usadas sólo por cierto tipo de software.
- Avanzado e investigación. Prácticas innovadoras que han sido probadas y usadas por sólo algunas organizaciones y conceptos que están aún siendo desarrolladas y probadas en organizaciones de investigación.
- Generalmente aceptado. Prácticas establecidas ampliamente y recomendadas por muchas organizaciones.

[15] Para más información puede dirigirse a la siguiente dirección: www.swebok.org

El documento se divide en diez áreas de conocimiento más algunos capítulos de apoyo. Cada área de conocimiento cuenta con una introducción, conceptos más relevantes y temas relacionados. Además, cada área de conocimiento tiene una amplia bibliografía y los temas son referenciados hacia libros o documentos en específico.

Las áreas de conocimiento son:

- Requerimientos de software
- Diseño de software
- Construcción de software
- Pruebas de software
- Mantenimiento de software
- Administración de la configuración de software
- Administración de la ingeniería de software
- Proceso de ingeniería de software
- Herramientas y métodos de la ingeniería de software
- Calidad del software

También se establecen límites entre la ingeniería de software y otras disciplinas que participan de cierta forma con ella. Se sugiere tener un cierto conocimiento de estas disciplinas relacionadas pero no es uno de los objetivos de SWEBOK.

Disciplinas relacionadas:

- Ciencias de la computación
- Ingeniería en computación
- Administración
- Matemáticas
- Administración de proyectos
- Administración de la calidad
- Ergonomía del software
- Ingeniería de sistemas

Este documento cubre lo que sería el conocimiento necesario, pero no el suficiente para un ingeniero en software.

PMBOK

PMBOK es el conocimiento acumulado dentro de la profesión de administración de proyectos. Elaborado por el PMI; el cual lo utiliza como parte de sus fundamentos. Es un estándar que se mantiene en constante evolución.

Incluyen prácticas que han sido ampliamente aplicadas y probadas, así como prácticas innovadoras que surgen de la profesión, incorporan material publicado e inédito. También incluye un léxico elemental para la profesión.

El propósito de la guía del PMBOK es identificar un subconjunto que es generalmente reconocido como buenas prácticas. Identificar significa proporcionar una visión general al contrario de una descripción exhaustiva. Generalmente reconocido significa que el conocimiento y las prácticas descritas son aplicables a muchos proyectos y que existe un amplio consenso acerca de su valor y utilidad. Las buenas prácticas se refieren a que existe un acuerdo de que la correcta aplicación de las habilidades, herramientas y técnicas pueden mejorar las oportunidades las oportunidades de éxito en un amplio rango de diferentes proyectos.

El estándar documenta la información necesaria para iniciar, planificar, ejecutar, monitorear y controlar y cerrar un solo proyecto; identifica los procesos de administración de proyectos que han sido reconocidos como buenas prácticas.

"Esto no quiere decir que el conocimiento, habilidades y procesos descritos deben de ser aplicados siempre de manera uniforme en todos los proyectos. El administrador de proyectos, en colaboración con su equipo, es responsable de determinar qué procesos son apropiados y el grado de rigor apropiado para cada proceso para cualquier proyecto."[16]

Los cuarenta y cuatro procesos se encuentran dispersos en una matriz de nueve áreas de conocimiento y cinco grupos de procesos.

Las áreas de conocimiento son áreas identificadas por la administración de proyectos y descritas en términos de sus procesos, prácticas, entradas, salidas, herramientas y técnicas.

Áreas de conocimiento:

- Integración de la administración del proyecto (*project management integration*)
- Administración del alcance del proyecto (*project scope management*)
- Administración de la duración del proyecto (*project time management*)
- Administración de los costos del proyecto (*project cost management*)
- Administración de la calidad del proyecto (*project quality management*)
- Administración de recursos humanos del proyecto (*project human resource management*)
- Administración de la comunicación del proyecto (*project communications management*)
- Administración de riegos del proyecto (*project risk management*)
- Administración del abastecimiento del proyecto (*projecto procurement management*)

Los grupos de procesos no son las fases de un proyecto, ya que en proyectos grandes o complejos los grupos de procesos podrían encontrarse en distintas fases. Los grupos de procesos son requeridos para cualquier proyecto, tienen dependencias internas y deben desempeñarse en la misma secuencia en cada proyecto.

Grupos de procesos:

- Inicial
- Planificación
- Ejecución
- Monitoreo y control
- Cierre

[16] Project Management Institute. *Project management Body of Knowledge*. Pennsylvania, PMI, 2004, p. 37.

Pruebas de software

Las pruebas de software representan una parte importante para asegurar la calidad del software, a través de ellas se corroboran o desechan el cumplimiento de los elementos que integran o que debería de ser parte del software. Existe una extensa variedad de pruebas que pueden aplicarse, muchas de ellas se agrupan según su ámbito de aplicación.

Anteriormente las pruebas de software se colocaban dentro de una de las fases del ciclo de desarrollo de software, actualmente las pruebas de software han dejado de ser vistas como una fase al final de la codificación con el fin de encontrar defectos en fases más tempranas, por lo que ahora se ve involucrado a lo largo del proceso de desarrollo y mantenimiento. De esta manera se adopta una postura preventiva más que de corrección porque le permite al equipo anticiparse a la aparición de los defectos.

Estudios realizados por IBM demostraron que el software durante el ciclo de vida de desarrollo produce 60 defectos. El estudio también mostró que la efectividad de las pruebas anteriores a la codificación es del 50 por ciento y del 80 por ciento después de la codificación. Este estudio en conjunto con otros mostró que por lo menos es 10 veces más costoso corregir un defecto después de la codificación y 100 veces más costoso corregirlo una vez que se encuentra en producción[17] (Fig. 6).

Los costos se incrementan claramente debido al nivel de esfuerzo y recursos requeridos para realizar una corrección. Un defecto podría implica modificación del código, los manuales, la planificación de las actividades y quizá algunos aspectos de la arquitectura, lo que requiere tiempo extra y tal vez más recursos para solucionar el defecto y cumplir con las otras tareas asignadas previamente que se deben de cumplirse. Todo esto tiene un impacto notable al momento de traducirlo a cifras en dinero.

Desarrollo normal		Fase	Costo del defecto	Desarrollo con pruebas en paralelo	
Costo acumulado	**Defectos acumulados/1000 líneas de código**			**Defectos acumulados/1000 líneas de código**	**Costo acumulado**
0	20	Requerimientos 20 defectos	1	10	10
0	40	Diseño 20 defectos	1	15	25
0	60	Código 20 defectos	1	18	42
480	12	Pruebas 80% de reducción de defectos	10	4	182
1680	0	Producción “0” defectos	100	0	584

Fig. 6 Costos de las pruebas en las fases de desarrollo de software

Dentro de la actividad de las pruebas de software encontramos algunas definiciones básicas que serán de utilidad para desenvolvernos en ella. Mucha de esta terminología se encuentra en el estándar IEEE 610.12 – 1990. A continuación se mencionan brevemente algunos de estos términos básicos.

Error. La gente comete errores o equivocaciones. Los errores tienden a propagarse; un error en la etapa de requerimientos puede acrecentarse durante el diseño y aumentar aún más durante la codificación.

[17] Perry, William E. *Effective methods for software testing*. New York, Wiley, 1995, p. 16.

Defecto. El defecto es el resultado de un error. De forma más precisa podría decirse que un defecto es la representación de un error, representación que queda expresado en forma de texto, diagrama de flujo, código fuente, etc. Los sinónimos utilizados son *bug* y falta. Los defectos pueden ser difíciles de detectar. Cuando se tiene una omisión el defecto consiste en que algo que debería estar presente no lo está. Por lo que podría decirse que existen dos tipos de defectos: por omisión y por comisión. El defecto por comisión tiene que ver con una representación incorrecta de lo solicitado. El defecto por comisión ocurre al momento de ingresar información. De estas dos el defecto por omisión es más difícil de detectar y resolver.

Falla. La falla aparece al momento en que se ejecuta un defecto. Este por lo tanto sólo se da para una representación ejecutable y en defectos por comisión. Entonces, ¿qué hacer con los defectos que nunca se ejecutan o no se ejecutan por mucho tiempo? Una opción aceptable es realizar buenas revisiones, que pueden prevenir fallas y encontrar defectos por omisión.

Casos de prueba. Estructura que permita probar todos los procesos posibles del sistema para encontrar sus inadecuaciones con los requerimientos y normas establecidas, con el menor esfuerzo y tiempo posibles. Un conjunto de entradas, condiciones de ejecución y resultados esperados desarrollados para un objetivo particular como, por ejemplo, ejercitar un camino concreto de un programa o verificar el cumplimiento de un determinado requisito.

Verificación. Son todas las actividades a través del ciclo de vida que aseguran que los productos internos en el proceso cumplan con sus especificaciones.

Validación. Asegura que el producto final cumple con las especificaciones establecidas.

Pruebas estáticas. Son las realizadas sin la ejecución del código.

Pruebas dinámicas. Son realizadas con la ejecución del código.

Pruebas funcionales. Pruebas a partir de los requerimientos (qué es lo que se supone que hace). Se asegura de que los requerimientos sean cumplidos adecuadamente por el software.

Pruebas estructurales. Pruebas sobre la estructura del sistema (cómo el sistema fue implementado). Se asegura de que el producto diseñado es estructuralmente lógico.

Pruebas manuales. Son pruebas realizadas por personas.

Pruebas automatizadas. Son pruebas realizadas por la computadora.

Pruebas de caja negra. Pruebas basadas en especificaciones externas sin el conocimiento previo de cómo se encuentra construido.

Pruebas de caja blanca. Pruebas basadas en el conocimiento de la estructura interna y lógica.

Para este proyecto se entiende que las pruebas de software son la actividad encargada de obtener la evidencia suficiente entre lo requerido y lo realizado para propósitos establecidos.

Lo requerido se refiere tanto a los requerimientos explícitos del cliente y/o usuario como los requerimientos implícitos que pueden ser ignorados o asumidos como obvios por parte del cliente y/o usuario. Lo realizado se refiere a todo aquel producto generado durante el desarrollo de software.

A través de las pruebas se pondrá en evidencia la correspondencia entre lo que se ha solicitado y lo que se ha generado.

Métricas

Introducción a la medición

Las mediciones no son de uso exclusivo de los sistemas de información, sino que le acompañan de cierta forma al hombre desde sus inicios. La medición es una actividad que aparece constantemente en la vida del hombre y le permite comprender el mundo, interactuar con su entorno y mejorar las cosas que hace.

La medición es resultado del refinamiento del sentido común que utiliza el hombre para interactuar con el mundo de manera cotidiana: la comparación. La comprensión del hombre sobre las cosas tiende a ser por medio de la comparación. Se comienza comprendiendo el mundo partiendo de relaciones menos sofisticadas que no requieren de herramientas de medición más que el sentido común.

Cada uno de los aspectos que medimos es sometido a comparación. Al medir capturamos las características de una entidad, que nos permitirán diferenciarla de otras entidades.

Uno de los objetivos de la ciencia es encontrar la forma de medir los atributos de las cosas que le son de interés para explicar su comportamiento. La medición hace conceptos más claros y por lo tanto más comprensibles y controlables.

La representación de los atributos se lleva a cabo por medio de números o símbolos. Estas representaciones son abstracciones de lo que observamos en el mundo y tratan de preservar la relación que se observa entre las entidades. De esta forma es posible elaborar juicios de las características que mantienen las entidades.

La teoría de la medición busca formalizar aquella intuición que permite conocer cómo trabaja el mundo. Los datos obtenidos de las mediciones deberían de representar los atributos de las entidades que observamos y la manipulación de esos datos debería de conservar las relaciones que observamos entre las entidades. Establece las reglas que deben llevarse a cabo para las mediciones, que son las que le proporcionan consistencia e interpretación de los datos. Nos dicen cuándo y cómo medir.

Una sugerencia para elaborar mediciones formales nos indica llevar a cabo los siguientes pasos:

- Identificar los atributos de una entidad en la realidad.
- Identificar una relación empírica para los atributos.
- Identificar las relaciones numéricas correspondientes a cada relación empírica.
- Definir el mapeo de las entidades del mundo real a números.
- Revisar que las representaciones numéricas se conserven y que sean preservadas por las relaciones empíricas.

Importancia de las métricas en el desarrollo de software

La ausencia de medición en la ingeniería de software se debe a la falta de un riguroso acercamiento que se tiene a ésta a comparación de otras ingenierías. Provocando que las mediciones sean dispersas, inconsistentes e incompletas. Esto es en gran parte a las dificultades a las que se deben enfrentar con los objetos de medición en el software, caso contrario con las ciencias físicas en las que existen mayores acuerdos y desarrollos en instrumentos de medición.

Las ventajas de las métricas se pueden observar por su utilidad, ya sea para mejoran el proceso, entender causas, etc. Para que las ventajas sean evidentes dependerá en gran parte de la definición de los objetivos.

Aún con las dificultades es posible encontrar ventajas que ofrecen las mediciones en el desarrollo y mantenimiento de software:

- Hay mediciones que permiten entender lo que está pasando durante el proceso de desarrollo y mantenimiento a partir de una línea base.
- Hay mediciones que nos permiten controlar los proyectos de desarrollo de software.
- Permiten mejorar los procesos y productos.
- Establecer un programa de métricas mejora el entendimiento del software (producto) y del proceso. El entendimiento conlleva una mejor administración y mejoras en el proceso.
- Se pueden utilizar con el fin de determinar la calidad.
- Medir un proceso lo hace más visible y por lo mismo más entendible y controlable.
- Las métricas permiten unificar criterios acerca de distintos aspectos.
- Las mediciones proveen un mecanismo cuantificable de las características.
- Nos permiten estimar: costos, tamaño, esfuerzo, productividad, etc.
- Nos proporcionan los elementos para tomar mejores decisiones.

Para la NASA existen tres razones principales para realizar actividades de medición en el proceso de desarrollo y mantenimiento de software:

- Incrementar el entendimiento en procesos, modelos y productos de software.
- Administrar proyectos de software.
- Servir de guía para mejorar los procesos en la ingeniería de software.

Para Pressman existen las siguientes razones para realizar mediciones en el software:

- Para indicar la calidad del producto.
- Evaluar la productividad de la gente.
- Para evaluar los beneficios por utilizar nuevos métodos y herramientas.
- Establecer una línea base para realizar estimaciones.
- Como justificación para nuevas herramientas o formación adicional.

Definiciones

A continuación se presentan las definiciones básicas que son comúnmente utilizadas en las actividades relacionadas a la medición.

Una entidad es un objeto o evento en el mundo real. Un atributo es una característica o propiedad de una entidad.

Métrica. Es el planteamiento o descripción de lo que se quiere medir, tiene asociado un proceso de medición y como consecuencia una medida.
Métrica. Es una propiedad que se puede medir que sirve como indicador de uno o más de los criterios de calidad.

Medición. Es el proceso mediante el cual números o símbolos se asignan a atributos de entidades del mundo real, de tal forma que los describen de acuerdo a reglas definidas claramente.
Medición (IEEE 1990). Una actividad que comprueba o evalúa por medio de la comparación de un estándar.

Medida. Es el resultado de hacer una medición.

Atributos internos. Son aquellos que pueden ser medidos en términos del propio objeto. Es decir, un atributo interno puede ser medido mediante el examen del objeto, separado de su comportamiento.

Atributos externos. Son aquellas que pueden ser medidas solamente con respecto a cómo el objeto se comporta en su ambiente.

Contar. Medición que utiliza el número cardinal de los elementos de un conjunto proporcionando de esta manera una medida exacta.

Medida directa. Usualmente es un proceso visual que consiste en hacer una comparación directa del objeto con una adecuada unidad de medida estándar.

Medida indirecta. Debido a que no se pueden realizar de manera directa se utilizan instrumentos de medida indirecta para registrarlos sobre una escala.

Medición subjetiva. Depende del ambiente en el que se realiza la medición y variará dependiendo de la persona que realice la medición. Reflejan un juicio personal y es difícil alcanzar un consenso.

Medición objetiva. Es cuando puede realizarse por diferentes personas y obtener la misma medición, logrando con esto una consistencia en el resultado.

Tipos de métricas

Clasificar las mediciones nos permite organizar las métricas a un enfoque en específico. A continuación se presentan algunas de estas agrupaciones.

Algunos tipos de métricas que podemos encontrar son:

- Métricas para costos
- Métricas para errores
- Métricas para las características del proceso
- Métricas para la dinámica del proyecto
- Métricas para las características del proyecto
- Métricas básicas o primitivas
- Métricas calculadas
- Métricas para la administración de proyectos
- Métricas para la administración de un proceso
- Métricas del producto
- Métricas por objetivo del negocio
- Métricas de control
- Métricas para realizar pronóstico
- Métricas de productividad
- Métricas técnicas
- Métricas de calidad
- Métricas orientadas al tamaño
- Métricas orientadas a la función
- Métricas orientadas a la persona

Cada una de estas clasificaciones no son de uso exclusivo del software, sino que pueden utilizarse en otras actividades y, además, una métrica puede aparecer en más de una categoría.

Cada perspectiva cuenta con sus propios objetivos y éstos determinan cuáles son las métricas más importantes para ellos.

Consideraciones para las métricas

La implementación y uso de métricas tiene la misma importancia que otros elementos por lo que no deben dejarse de lado.

En una buena planificación de métricas es conveniente considerar algunas cuestiones como las siguientes:

- ¿Cuáles son las métricas que serán utilizadas?
- ¿Quiénes realizarán las actividades relacionadas con las métricas?
- ¿En qué momento del proceso se realizarán las actividades de medición?
- ¿Qué información será recolectada?
- ¿De qué forma los datos serán analizados y presentados?
- ¿A todos los usuarios se les presentará la misma información?
- ¿Cuál es el costo de implementar y utilizar las métricas?

Resulta una ventaja contar con un número reducido de métricas que se encuentren alineadas a objetivos concretos. Ya que la aplicación de cada métrica requiere un esfuerzo determinado. No por tener mayor número de métricas el beneficio es mayor. Por lo que es recomendable que los reportes cuenten con sólo la información necesaria. La recolección de datos es una pequeña parte del proceso de medición.

Para la NASA,[18] de acuerdo a la información recolectada y analizada de 17 años acerca de los costos en las actividades de medición durante los proyectos de desarrollo y mantenimiento de software sugieren que:

- El costo de medición en un proyecto de desarrollo y mantenimiento no debe ser mayor al 2% del presupuesto total del proyecto.
- El costo de soporte técnico no debe de encontrarse del 3% al 7% del presupuesto total del proyecto.
- El costo de análisis y empaquetado puede ir del 5% al 15% del presupuesto total del proyecto.

Selección de métricas

Para implementar de manera exitosa un conjunto de métricas es necesario seleccionar aquellas que sean importantes.

Para el SEL de la NASA no hay mediciones universales para mejorar, sino que considera que sólo las condiciones particulares de cada organización establecen el conjunto de métricas que le son de importancia.

Para utilizar métricas es importante contar con un conjunto definido de objetivos. La recolección de datos es importante, pero no es el objetivo.

Pasos para establecer un programa de medición:

[18] Software Engineering Laboratory. *Software measurement guidebook*. Greenbelt, National Aeronautics and Space Administration, 1995, p. 31.

- Establecer las metas de la organización, éstas deben reconocer la necesidad de establecer un programa de medición para alcanzar las metas.
- La medición debe estar sustentada en un objetivo claramente definido y fácil de entender. Los objetivos variarán dependiendo del rol que finalmente hará uso de la información.

Las métricas por seleccionar deberían de contar con algunas características como las que se sugieren a continuación.

Para Yourdon, una métrica debe de ser:
- Entendible
- Ampliamente probada
- Económica
- Tener un alto nivel de apalancamiento
- Oportuna

Para Watts sugiere siete criterios para una métrica
- Objetividad
- Confiabilidad
- Validez
- Estandarización
- Facilidad de comparación
- Economía
- Utilidad

GQM es una opción para seleccionar métricas. Meta-Pregunta-Métrica (GQM por sus siglas en inglés) es un método creado por V. Basili y D. Weiss (1984) y extendido posteriormente por D. Rombach (1990). Presenta de manera sistemática un acercamiento de las metas generales con las más específicas para posteriormente establecer métricas posibiliten el logro de las metas. De esta manera se asegura que sean útiles y que tendrán el impacto necesario en las metas generales. Estas metas generales no son necesariamente los de la organización en general, sino que pueden ser los de un departamento o algo más específico.

A partir de la definición de metas se aplica el método para refinarlas en preguntas y posteriormente en métricas, las cuales proveen de la información necesaria para contestar las preguntas. El método provee un plan de medición que trata con un conjunto de problemas particulares y un conjunto de reglas para obtener la interpretación de los datos. Esa interpretación proporciona las elementos necesarios que indiquen si las metas fueron alcanzadas.

Los beneficios que podemos encontrar con el método son:
1. Alinear las Métricas con los negocios de la organización y las metas técnicas.
2. Mejorar el proceso del software
3. Gerenciar el riesgo
4. Mejorar la calidad del producto

El método consta de cuatro fases:
1. Fase de planificación, en el cual el proyecto es seleccionado, definido, caracterizado y planificado.
2. Fase de definición, durante el cual el programa de medición es definido (meta, pregunta, métricas e hipótesis son definidas) y documentado.
3. Fase de colección de datos, durante la colección actual de datos se lleva a cabo.
4. Fase de interpretación, los datos son procesados para contestar las preguntas planteadas y determinar si las metas fueron alcanzadas.

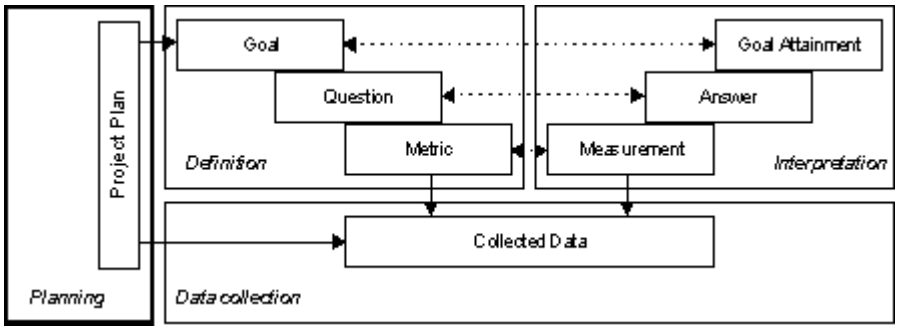


Fig. 7 Las cuatro fases del método GQM

Planificación. Los objetivos primarios de esta fase es conjuntar toda la información necesaria para un inicio exitoso y para prepara y motivar a los miembros de una organización para un programa de medición. El plan del proyecto es un entregable importante en esta fase. En ella se documenta el procedimiento, calendario, capacitación y objetivos de un programa de medición, proporcionando una base para su administración.

Esta fase se compone además de cinco pasos:
1. Establecer el equipo de GQM
2. Seleccionar el área de mejora
3. Seleccionar el proyecto y establecer el equipo del proyecto
4. Crear el plan del proyecto
5. Entrenar y promocionar

Definición. Se refiere a todas las actividades que deberán de realizarse para que formalmente se defina el programa de medición. Durante la fase se producen tres entregables:
- Plan GQM
- Plan de medición
- Plan de análisis

Esta fase consta de once pasos:
1. Definición de las metas de medición
2. Revisar o elaborar los modelos del proceso de software
3. Conducir las entrevistas del GQM
4. Definir las preguntas y las hipótesis
5. Revisar las preguntas y las hipótesis
6. Definir métricas
7. Revisar la consistencia de las métricas y que se encuentren completas
8. Elaborar el plan de GQM
9. Producir el plan de medición
10. Producir el plan de análisis
11. Revisar los planes

Colección de datos. Corresponde a las herramientas para realizar el registro. Incluye la manera en la que el procedimiento es definido, la manera en la que las formas son aplicadas y la manera en la cual las herramientas apoyan la colección de datos.

Para la colección de datos se sugieren cinco pasos:
1. Periodo de prueba
2. Sesión de inicio
3. Crear métricas base
4. Almacenar y revisar datos
5. Almacenar los datos de la medición en una métrica base

6. Definir hojas de análisis y láminas de presentación

Interpretación. Esta fase incluye todas las actividades requeridas para realmente obtener, almacenar y procesar los datos de las mediciones. Se enfoca en la elaboración de conclusiones del programa de medición. Conclusiones que son específicas para cada programa de medición.

Los resultados del programa de medición son discutidos en sesiones de retroalimentación. Se sugieren tres pasos para organizar esta sesión:

1. Preparar la sesión de retroalimentación
2. Organizar y mantener la sesión de retroalimentación
3. Reportar los resultado de las mediciones

El resultado del método es la especificación de un programa de medición apuntando a un conjunto de cuestiones particulares y un conjunto de reglas para interpretar los datos.

Un principio importante del método es que las mediciones deben de estar orientadas hacia las metas. Es importante definir las metas para transformarlas en actividades que puedan ser medidas durante la ejecución del proyecto. De esta forma la GQM define las métricas de arriba hacia abajo y analiza e interpreta los datos de abajo hacia arriba.

El modelo GQM comienza de arriba hacia abajo definiendo las metas, que posteriormente se convertirán en preguntas que después se refinarán en métricas que pretenden proporcionar la información para contestar las preguntas. Mientras la medición e interpretación viene de abajo hacia arriba. Como las métricas fueron definidas desde un principio, las mediciones deben de ser interpretadas y analizadas con respecto a esas metas para saber si éstas fueron alcanzadas.

Es recomendable que las metas, preguntas y las métricas sean elaboradas por los expertos en la organización.

Capitulo III Propuesta

Este capítulo presenta la selección de elementos con los que se pretende cubrir los objetivos planteados al inicio de este trabajo. Selección que pretende ser lo más objetiva posible y que para lograrlo se apoya en las características de la organización.

Las opciones presentadas en el capítulo anterior permiten mostrar algunas de buenas prácticas que existen en la industria para mejorar algún aspecto del desarrollo de software. Hasta este punto el obstáculo no es encontrar las posibles soluciones que se encuentren disponibles, sino seleccionar de ese número considerable de opciones aquellas que sean significativas para lograr los objetivos de este proyecto.

Es por ello que la delimitación de los elementos se encuentra en relación directa con las necesidades, antecedentes y restricciones de la organización. Estos elementos son los que permiten acotar de manera importante la dimensión de posibilidades con las que se cuenta. De esta forma se cubren los tres objetivos que tienen que ver con la problemática de la jefatura.

Objetivo 1

Ordenar el desarrollo de software, es el primer objetivo. Para lograrlo el primer elemento a seleccionar es el modelo de referencia. Un modelo proveerá la base sobre la cual se organizan los demás elementos, estableciendo el qué, quién, cómo y el cuándo de cada uno de ellos. De esta forma se establece un orden e interacción en el desarrollo de software.

Para seleccionar un modelo de referencia se tomaron en cuenta un conjunto de exigencias mínimas establecidas por parte de la organización que el modelo debe de cumplir. El modelo de referencia seleccionado debe de cumplir con el mayor número de las exigencias establecidas, porque de esa manera refleja su adecuación a la organización en la que será implementado.

A continuación se describen las exigencias solicitadas al modelo de referencia:

- Específico para el área. El modelo debe de ser específico para las actividades de desarrollo y mantenimiento de software que se realizan.
- Implantación gradual. Debe de proporcionar una guía de implantación gradual de las sugerencias del modelo de referencia.
- Detallado. Debido a es el primer modelo de referencia consultado y las personas no cuentan con profundos conocimientos relacionados a modelos de calidad, se requiere de una detallada descripción del modelo de referencia.
- Aplicable a organizaciones pequeñas. Debido a que la jefatura cuenta con pocos integrantes es importante que el modelo cuente con las tareas y los roles que posibiliten la ejecución sin tener que recurrir a más personal del que se cuenta.
- Costo aceptable. Una de las restricciones con las que se cuenta se refiere a los recursos monetarios y esto establece que se busque un modelo que requiera pocos recursos para implantarlo. Esto incluye cursos, material de consulta, asesorías, certificaciones, etc.
- Claridad. El modelo debe de contar con un lenguaje fácil de entender que evite ambigüedades al momento de interpretar el modelo de referencia.
- Software de apoyo. Es importante que el modelo cuente con software de apoyo que permita automatizar algunas tareas sugeridas o relacionadas al modelo.
- Referencias. El modelo debe de contar con referencias acerca de las experiencias que se han generado por la utilización del modelo y que de alguna forma pueden orientar en cierto momento en la implementación del modelo.

Modelo / Requerimiento	ISO 9001:2000	CMMI	ISO 15504	MoProSoft
Específico para el área	No	Sí	Sí	Sí
Implantación gradual	No	Sí	Sí	Sí
Detallado	No	No	No	Sí
Aplicable a organizaciones pequeñas	No	No	No	Sí
Costo aceptable	No	No	No	Sí
Claridad	No	No	No	Sí
Software de apoyo	Sí	Sí	No	No
Referencias	Sí	Sí	Sí	Sí

Fig. 1 Tabla de comparación de procesos

Como resultado de la comparación, MoProSoft versión 1.3 (coloreado) representa una opción asequible para la organización porque: es una norma mexicana diseñada para el desarrollo y mantenimiento de software, cuenta con la flexibilidad para que pueda ser adoptado por áreas internas, muestra los niveles de capacidad de procesos para que pueda realizarse una implantación gradual del modelo de referencia conforme vaya madurando la organización, cuenta con un nivel de detalle adecuado para la experiencia de la organización, las tareas y sugerencias realizadas por el modelo pueden ser realizadas por un número reducido de personas, los costos de implantación del modelo son inferiores a los otros modelos, el modelo se encuentra escrito completamente en español desde su creación y utiliza un lenguaje claro, cuenta con referencias sobre su implantación y foros de discusión. El modelo cuenta con cursos, evaluaciones y un organismo (NYCE) encargado de verificar la aplicación del modelo en las organizaciones que lo requieran. Uno de los inconvenientes por su reciente creación es que aún no cuenta con software de apoyo.

MoProSoft representa una opción aceptable también para las organizaciones que pretendan evaluarse en modelos como ISO 9001:2000 y CMM v1.1, ya que tiene un nivel de compatibilidad con ambos modelos[1].

De los procesos que comprende MoProSoft sólo se tomarán en cuenta los siguientes:

- Administración de proyectos específicos
- Desarrollo y mantenimiento de software

Estos dos procesos se orientan de manera directa a los intereses de la organización, por lo que los demás procesos no entran por el momento en los planes de implantación, además de que por el momento la organización no busca algún tipo de certificación del modelo. Por tratarse de una jefatura dentro del área de sistemas los demás procesos mencionados por el modelo no forman parte de sus funciones primarias porque son realizadas por otras áreas dentro de la empresa. Aunado a esto cabe recordar que el giro de la empresa no es el desarrollo de software y el software que se desarrolla es utilizado como apoyo a sus actividades primarias.

Ambos procesos son implantados en nivel dos (de cinco que menciona el modelo). Esto es porque ya se cuenta con antecedentes que implican de cierta forma gran parte de las actividades del nivel uno a las que se refiere el modelo, con lo que basta con hacer sólo unas adecuaciones con lo ya

[1] Para obtener el nivel de compatibilidad entre los modelos consulte Oyvind, Mo. *Comparación del modelo de procesos para la industria de software (MoProSoft) con las normas y modelos de referencia*. Tesis para obtener el grado de maestro en ciencias e ingeniería en computación. México, UNAM, 2005.

se encuentra disponible, no se aplica un nivel más avanzado porque para ello se requiere de datos históricos y experiencias generadas por el modelo, cosas que no se han generado por el momento.

A continuación se presenta unas tablas con los procesos seleccionados en los niveles 1 y 2 con sus correspondientes tareas[2]. Contiene en sus primeras tres columnas (de izquierda a derecha) las sugerencias del modelo y las siguientes cuatro columnas contienen, en caso de ser requerido por la organización, los elementos y ajustes pertinentes para llevar a cabo las sugerencias del modelo. Cada uno de los números que se ven involucrados en la columna Métodos, técnicas y herramientas son elementos recolectados de distintas fuentes (SWEBOK y PMBOK) e incorporados al modelo de referencia para complementarlo. Una breve descripción de estos últimos elementos puede ser consultada en el Anexo de este documento. Los elementos que no apliquen se les coloca las siglas NA (no aplica).

Rol	No. tarea	Productos involucrados	Requerido	Métodos, Técnicas y herramientas	Formatos involucrados	Ajustes
ET	A1.1	Plan de desarrollo	Sí	NA	Plan del proyecto	El Plan de desarrollo es sustituido por el Plan del proyecto
RDM AN	A2.1	Plan de desarrollo	Sí	NA	Plan del proyecto	El Plan de desarrollo es sustituido por el Plan del proyecto
AN CL US DU	A2.2	Especificación de requerimientos	Sí	NA	Especificación de requerimientos	NA
RM	A2.10	Manual de usuario	No	NA	NA	No se elaborará un Manual de usuario preeliminar
RDM	A2.13	Especificación de requerimientos Manual de usuario Configuración de software	Sí	NA	Especificación de requerimientos	NA
RDM AN DI	A3.1	Plan de desarrollo	Sí	NA	Plan del proyecto	El Plan de desarrollo es sustituido por el Plan del proyecto
AN DI DU	A3.2	Análisis y diseño Especificación	Sí	NA	Análisis y diseño Especificación	NA

[2] Para obtener el detalle de cada tarea, las abreviaturas de los roles y de más elementos que comprende cada proceso consulte MoProSoft versión 1.3

Rol	No. tarea	Productos involucrados	Requerido	Métodos, Técnicas y herramientas	Formatos involucrados	Ajustes
		de requerimientos			de requerimientos	
RDM	A.3.10	Análisis y diseño Configuración de software	Sí	NA	Análisis y diseño	NA
RDM	A4.1	Plan de desarrollo	Sí	NA	Plan del proyecto	El Plan de desarrollo es sustituido por el Plan del proyecto
PR	A4.2	Componente de software Análisis y diseño	Sí	NA	Análisis y diseño	NA
RDM	A4.5	Componente de software Configuración de software	Sí	NA	NA	NA
RDM	A5.1	Plan de desarrollo	Sí	NA	Plan del proyecto	El Plan de desarrollo es sustituido por el Plan del proyecto
PR RPU	A5.2	Componentes	Sí	NA	NA	NA
RM	A5.3	Manual de operación	Sí	NA	NA	No se cuenta con un formato que defina su contenido en todos los casos
RM	A5.8	Manual de usuario	Sí	NA	NA	No se cuenta con un formato que defina su contenido en todos los casos
RDM	A5.11	Software Manual de operación Manual de usuario Configuración de software	Sí	NA	NA	NA

Fig. 2 Nivel 1 Desarrollo y mantenimiento de software

Rol	No. tarea	Productos involucrados	Requerido	Métodos, Técnicas y herramientas	Formatos involucrados	Ajustes
RGPY RAPE RDM	A1.1	Descripción del proyecto	Sí	NA	Descripción del proyecto	NA
RAPE CL	A1.3	Protocolo de entrega Descripción del proyecto	Sí	NA	Descripción del proyecto Plan del proyecto	NA
RAPE	A1.4	Descripción del proyecto Protocolo de entrega Ciclos y actividades	Sí	1, 2, 3, 4, 5, 6, 7, 8, 9	Descripción del proyecto Plan del proyecto	NA
RAPE	A1.5	NA	Sí	NA	Plan del proyecto	NA
RAPE	A1.6	Tiempo estimado	Sí	10, 11, 12, 13, 14, 15	Plan del proyecto	NA
RAPE	A1.7	Plan de adquisiciones y capacitación	Sí	16, 17	Plan del proyecto	NA
RGPY RAPE	A1.8	Equipo de trabajo Descripción del proyecto	Sí	NA	Plan del proyecto Descripción del proyecto	NA
RAPE	A1.9	Calendario	Sí	20, 21	Plan del proyecto	NA
RAPE	A1.10	Costo estimado	Sí	18, 19	Plan del proyecto	NA
RGPY RAPE RDM	A1.11	Plan de manejo de riesgos	Sí	22, 23, 24, 25	Plan del proyecto	NA
RAPE	A1.12	Plan del proyecto	Sí	NA	Plan del proyecto	NA
RAPE RDM	A1.13	Plan de desarrollo Plan del proyecto	No	NA	NA	El Plan de desarrollo es sustituido por el Plan del proyecto
RAPE RDM	A2.1	Equipo de trabajo	Sí	NA	Plan del proyecto	NA
RAPE CL	A4.1	Protocolo de entrega Plan del proyecto Documento de aceptación	Sí	NA	Plan del proyecto Documento de aceptación	NA

Fig. 3 Nivel 1 Administración de proyectos específicos

Rol	No. tarea	Productos involucrados	Requerido	Métodos, Técnicas y herramientas	Formatos involucrados	Ajustes
RDM	A1.2	Reporte de actividades	Sí	NA	Reporte de actividades	NA
RE	A2.3	Especificación de requerimientos Descripción del producto Reporte de verificación	Sí	NA	Especificación de requerimientos Plan del proyecto Reporte de verificación Verificación de Especificación de requerimientos	El Plan de desarrollo es sustituido por el Plan del proyecto
AN DU	A2.4	Especificación de requerimientos Reporte de verificación	Sí	NA	Especificación de requerimientos Reporte de verificación	NA
CL US RPU	A2.5	Especificación de requerimientos Reporte de validación	Sí	NA	Especificación de requerimientos Reporte de validación Validación de especificación de requerimientos	NA
AN DU	A2.6	Especificación de requerimientos Reporte de validación	Sí	NA	Especificación de requerimientos Reporte de validación	NA
RPU AN	A2.7	Plan de pruebas de sistema	Sí	NA	Plan de pruebas de sistema	NA
RE	A2.8	Plan de pruebas de sistema Especificación de requerimientos Reporte de verificación	Sí	NA	Plan de pruebas de sistema Especificación de requerimientos Reporte de verificación Verificación de Plan de pruebas de sistema	NA
RPU	A2.9	Plan de pruebas de sistema Reporte de verificación	Sí	NA	Plan de pruebas de sistema Reporte de verificación	NA
RE	A2.11	Manual de	No	NA	NA	No se

Rol	No. tarea	Productos involucrados	Requerido	Métodos, Técnicas y herramientas	Formatos involucrados	Ajustes
		usuario Especificación de requerimientos Reporte de verificación				elaborará un Manual de usuario preeliminar
RM	A2.12	Manual de usuario Reporte de verificación	No	NA	NA	No se elaborará un Manual de usuario preeliminar
RDM	A2.13	Plan de pruebas de sistema Configuración de software	Sí	NA	Plan de pruebas de sistemas	NA
RDM	A2.14	Reporte de actividades	Sí	NA	Reporte de actividades	NA
AN DI DU	A3.2	Registro de rastreo	Sí	NA	Registro de rastreo	NA
RE	A3.3	Análisis y diseño Registro de rastreo Especificación de requerimientos Reporte de verificación	Sí	NA	Análisis y diseño Registro de rastreo Especificación de requerimientos Reporte de verificación Verificación de Análisis y diseño Verificación de registro de rastreo	NA
AN DI DU	A3.4	Análisis y diseño Registro de rastreo Reporte de verificación	Sí	NA	Análisis y diseño Registro de rastreo Reporte de verificación	NA
CL RPU	A3.5	Análisis y diseño Reporte de validación	Sí	NA	Análisis y diseño Reporte de validación Validación de Análisis y diseño	NA
AN DI DU	A3.6	Análisis y diseño Reporte de validación	Sí	NA	Análisis y diseño Reporte de validación	NA
RPU	A3.7	Plan de pruebas	Sí	NA	Plan de	NA

Rol	No. tarea	Productos involucrados	Requerido	Métodos, Técnicas y herramientas	Formatos involucrados	Ajustes
		de integración			pruebas de integración	
RE	A3.8	Plan de pruebas de integración Análisis y diseño Reporte de verificación	Sí	NA	Plan de pruebas de integración Análisis y diseño Reporte de verificación Verificación de Plan de pruebas de integración	NA
RPU	A3.9	Plan de pruebas de integración Reporte de verificación	Sí	NA	Plan de pruebas de integración Reporte de verificación	NA
RDM	A3.10	Registro de rastreo Plan de pruebas de integración Configuración de software	Sí	NA	Registro de rastreo Plan de pruebas de integración	NA
RDM	A3.11	Reporte de actividades	Sí	NA	Reporte de actividades	NA
PR	A4.2	Análisis y diseño Registro de rastreo	Sí	26, 27, 28, 29, 30, 34, 37, 38, 39, 40, 41, 42, 43, 44.	Análisis y diseño Registro de rastreo	NA
RE	A4.3	Registro de rastreo Análisis y diseño Reporte de verificación	Sí	NA	Registro de rastreo Análisis y diseño Reporte de verificación Verificación de Registro de rastreo	NA
PR	A4.4	Registro de rastreo Reporte de verificación	Sí	NA	Registro de rastreo Reporte de verificación	NA
RDM	A4.5	Registro de rastreo Configuración de software	Sí	NA	Registro de rastreo	NA
RDM	A4.6	Reporte de actividades	Sí	NA	Reporte de actividades	NA
PR RPU	A5.2	Plan de pruebas de integración Reporte de	Sí	26, 27, 28, 29, 30, 34, 44.	Plan de pruebas de integración	NA

Rol	No. tarea	Productos involucrados	Requerido	Métodos, Técnicas y herramientas	Formatos involucrados	Ajustes
		pruebas de integración Registro de rastreo			Reporte de pruebas de integración Registro de rastreo	
RE	A5.4	Manual de operación Software Reporte de verificación	Sí	NA	Reporte de verificación Verificación de Manual de operación	NA
RM	A5.5	Manual de operación Reporte de verificación	Sí	NA	Reporte de verificación	NA
RPU	A5.6	Plan de pruebas de sistema Reporte de pruebas de sistema	Sí	26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 44.	Plan de pruebas de sistema Reporte de pruebas de sistema	NA
PR	A5.7	Reporte de pruebas de sistema	Sí	NA	Reporte de pruebas de sistema	NA
RE	A5.9	Manual de usuario Software Reporte de verificación	Sí	NA	Reporte de verificación Verificación de Manual de usuario	NA
RM	A5.10	Manual de usuario Reporte de verificación	Sí	NA	Reporte de verificación	NA
RDM	A5.11	Reporte de pruebas de integración Registro de rastreo Manual de usuario Configuración de software	Sí	NA	Reporte de pruebas de integración Registro de rastreo	NA
RDM	A5.12	Reporte de actividades	Sí	NA	Reporte de actividades	NA
RM	A6.1	Manual de mantenimiento	Sí	NA	NA	No se cuenta con un formato que defina su contenido en todos los casos
RE	A6.2	Manual de mantenimiento	Sí	NA	Reporte de verificación	NA

Rol	No. tarea	Productos involucrados	Requerido	Métodos, Técnicas y herramientas	Formatos involucrados	Ajustes
		Configuración de software Reporte de verificación			Verificación de Manual de mantenimiento	
RDM	A6.7	Reporte de actividades	Sí	NA	Reporte de actividades	NA

Fig. 4 Nivel 2 Desarrollo y mantenimiento de software

Rol	No. tarea	Productos involucrados	Requerido	Métodos, Técnicas y herramientas	Formatos involucrados	Ajustes
RAPE	A1.4	Ciclos y actividades	Sí	NA	Plan del proyecto	El Plan de desarrollo es sustituido por el Plan del proyecto
RAPE	A1.12	Plan del proyecto Solicitud de cambios Acciones correctivas o preventivas Acciones correctivas	Sí	NA	Plan del proyecto Solicitud de cambios Acciones correctivas o preventivas Acciones correctivas	No se utiliza Acciones correctivas o preventivas
RAPE RDM	A1.13	Plan de desarrollo Solicitud de cambios Acciones correctivas o preventivas Acciones correctivas	No	NA	NA	El Plan de desarrollo es sustituido por el Plan del proyecto
RAPE	A1.14	Plan del proyecto Plan de desarrollo Reporte de verificación	Sí	NA	Plan del proyecto Reporte de verificación Verificación de Plan del proyecto	El Plan de desarrollo es sustituido por el Plan del proyecto
RAPE	A1.15	Plan del proyecto Plan de desarrollo Reporte de verificación	Sí	NA	Plan del proyecto Reporte de verificación	El Plan de desarrollo es sustituido por el Plan del proyecto
RPGY	A1.16	Plan del proyecto Plan de desarrollo	Sí	NA	Plan del proyecto Descripción del proyecto	El Plan de desarrollo es sustituido por el Plan del

Rol	No. tarea	Productos involucrados	Requerido	Métodos, Técnicas y herramientas	Formatos involucrados	Ajustes
		Descripción del proyecto Reporte de validación			Reporte de validación Validación de Plan del proyecto	proyecto
RAPE	A1.17	Plan del proyecto Plan de desarrollo Reporte de validación	Sí	NA	Plan del proyecto Reporte de validación	El Plan de desarrollo es sustituido por el Plan del proyecto
RAPE RDM	A2.2	Plan de comunicación e implantación	No	NA	NA	NA
RAPE RDM	A2.3	Descripción del producto Equipo de trabajo Calendario	Sí	NA	Plan del proyecto	El Plan de desarrollo es sustituido por el Plan del proyecto
RAPE RDM RSC	A2.4	Plan de adquisiciones y capacitación Asignación de recursos	No	NA	NA	NA
RAPE RSC	A2.5	NA	No	NA	NA	NA
RAPE	A2.6	Reporte de actividades	Sí	NA	Reporte de actividades	NA
RAPE	A2.7	NA	Sí	NA	NA	NA
RAPE	A2.8	Registro de rastreo	Sí	NA	Registro de rastreo	NA
RAPE RDM	A2.9	Configuración de software	Sí	NA	NA	NA
RAPE RDM	A2.10	Solicitudes de cambios Plan del proyecto Plan de desarrollo	Sí	NA	Solicitudes de cambio Plan del proyecto	El Plan de desarrollo es sustituido por el Plan del proyecto
RAPE ET CL	A2.11	Minutas	Sí	NA	Minutas	NA
RAPE	A3.1	Plan del proyecto Plan de desarrollo Acciones correctivas	Sí	NA	Plan del proyecto Acciones correctivas	El Plan de desarrollo es sustituido por el Plan del proyecto
RAPE	A3.2	Plan de manejo de riesgos	Sí	NA	Plan del proyecto	NA
RAPE	A3.3	Reporte de seguimiento	Sí	NA	Reporte de seguimiento	NA

Rol	No. tarea	Productos involucrados	Requerido	Métodos, Técnicas y herramientas	Formatos involucrados	Ajustes
		Reportes de actividades			Reportes de actividades	
RAPE RSC	A4.2	NA	No	NA	NA	NA

Fig. 5 Nivel 2 Administración de proyectos específicos

Por cuestiones de espacio no se muestran los demás elementos sugeridos por el modelo que integran a cada proceso porque no se requirieron modificaciones relevantes y se llevarán a cabo como se indica.

Objetivo 2

Asegurar la calidad del desarrollo de software, es el siguiente objetivo que debe de cubrirse. Existen varias formas de lograrlo, pero específicamente este punto se refiere a los productos de software que se generan del desarrollo de software. El modelo de referencia seleccionado cuenta dentro de sus procesos con puntos específicos para el aseguramiento de la calidad de los productos. Sin embargo, los productos derivados de la construcción de software son a los que se refiere este apartado, las pruebas de software son la opción más adecuada para asegurar la calidad de estos productos. En este aspecto el modelo sugiere un conjunto de pruebas a realizar dentro de las actividades del proceso Desarrollo y mantenimiento de software, pero no indica los pasos y las técnicas que pueden utilizarse en cada una de ellas. Ante este vacío en las pruebas sugeridas por el modelo se pretende dotar a estas actividades con algunos elementos de apoyo proporcionados por las referencias del SWEBOK. La selección de pruebas recolecta aquellas que se ajustan a las sugerencias del modelo de referencia. Las pruebas se encargarán de corroborar el buen funcionamiento del software durante su desarrollo.

Para el proceso Desarrollo y mantenimiento de software, se tiene considerado la implantación de las pruebas de software de la siguiente manera.

Los tipos de pruebas de software que sugiere el modelo son: pruebas unitarias, pruebas de integración y pruebas de sistema. A cada grupo de pruebas se le asignan un conjunto de pruebas específicas que apoyan su realización.

A continuación se presenta una breve descripción de cada una de las pruebas que se incorporan al proceso.

Prueba de clases de equivalencia

Descripción
Es una técnica utilizada para disminuir el número de casos de pruebas a un nivel de manejo aceptable y manteniendo una cobertura razonable, ya que en vez de elaborar un caso de prueba a cada uno de los posibles valores de entrada se seleccionan los valores que equivaldrían a los demás valores de un grupo en específico. Se utiliza cuando hay varios valores de entrada con rangos establecidos por el manejo que se hace a estos rangos. Cada valor dentro de la clase es equivalente a las otras. Esto significa que con un valor de la clase se representarían todos los demás valores que componen a la clase de equivalencia y de esa manera se evita probar todos los valores. Para llevarlo a cabo se debe de tener las especificaciones que definan las clases de equivalencia. Llegando así a realizar un caso de prueba por cada clase de equivalencia. Para

clases de equivalencia compuestas por valores definidos se recomienda generar un caso de prueba por cada uno de estos valores.

Pasos para realizar la prueba
- Identificar las clases equivalentes
- Crear los casos de prueba para cada clase equivalente

Técnicas que utiliza
- Clases de equivalencia
- Casos de prueba

Prueba de valor límite

Descripción
Consiste en realizar pruebas con los valores que limitan a una clase de equivalencia para comprobar que el uso de estos valores sea el correcto conforme a la definición. Una clase de equivalencia puede, mas no es necesario que se encuentre junto a otra clase de equivalencia.

Pasos
- Identificar las clases de equivalencia
- Identificar los límites de cada clase de equivalencia
- Crear casos de prueba para cada valor límite

Técnicas que utiliza
- Clases de equivalencia
- Valor límite
- Casos de prueba

Prueba de tabla de decisión

Descripción
Se utiliza para analizar las reglas de negocio que se deben de implementar en un sistema y también se usan como un documento de diseño interno del sistema. También se utilizan como guía para crear casos de prueba ya que cada regla representada se puede convertir en por lo menos un caso de prueba.

Pasos
- Representar cada una de las reglas en la tabla de decisión
- Generar por lo menos un caso de prueba por cada regla representada

Técnicas que utiliza
- Tabla de decisiones
- Casos de prueba
- Clases de equivalencia
- Valor límite

Prueba en parejas

Descripción
Se dan casos en los que es necesario probar varias variables con varios valores para cada una de ellas, dando como resultado un escenario que genera **n** número de combinaciones posibles. Realizar y ejecutar un caso de prueba a cada una de estas combinaciones resulta inconveniente cuando la cantidad de éstas es muy elevada. Una forma de simplificar el número de casos de prueba es abarcando los casos en los que es más probable detectar. En esta prueba se especifican sólo las entradas de los casos de prueba.

Pasos
- Identificar los pares
- Crear casos de prueba

Técnicas que utiliza:
- Arreglos ortogonales
- Algoritmo de allpairs
- Casos de prueba

Prueba de transición de estados

Descripción
Es una herramienta que permite capturar cierto tipo de requerimiento de sistemas y para documentar el diseño interno de los sistemas. Los diagramas documentan los eventos que son ingresados y procesados por el sistema así como las respuestas del sistema. Las tablas de decisión son más enfocadas al procesamiento de reglas mientras que esta prueba se enfoca a la interacción del sistema.

Pasos
- Identificar las entidades específicas del sistema.
- Identificar estados, transiciones, eventos y acciones que realiza una entidad específica.
- Representar gráficamente todos los elementos detectados a un diagrama de transición de estados.
- Elaborar tabla de transición de estados.
- Elaborar casos de prueba.

Técnicas que utiliza
- Diagramas de transición de estados
- Tabla de transición de estados
- Casos de prueba de transición de estados

Prueba de análisis de dominio

Descripción
Se lleva a cabo para realizar una prueba con múltiples variables de manera simultánea. Debido principalmente al escaso tiempo que se tiene para crear casos de prueba individuales y a la interacción que existe entre variables, de esta forma es posible identificar defectos que no podrían alcanzarse con pruebas individuales. Esta es una técnica utilizada para identificar casos de prueba con múltiples variables que puede o deberían de probarse de manera conjunta. Construyendo de esta manera pruebas de equivalencia de clases y valor límite en *n* dimensiones simultáneas.

Pasos
- Identificar la variables
- Identificar los valores de las variables
- Identificar los valores límite de cada variable
- Identificar el tipo de condicional lógico es el que delimita al valor límite
- Asignar los puntos **On** y **Off** dependiendo del condicional lógico
- Llenar la matriz de dominio
- Crear casos de prueba

Técnicas que utiliza
- Clases de equivalencia
- Valor límite
- Matriz de dominio
- Análisis de dominio
- Casos de prueba

Pruebas de casos de uso

Descripción
Los casos de uso, creados por Ivar Jacobson, se utilizan para representar las transacciones que un sistema procesa. Un caso de uso es un escenario que describe el uso de un sistema por un actor para cumplir una meta en específico. Los casos de uso son definidos desde la perspectiva del usuario y no del sistema. Las definiciones internas del sistema no son parte de los casos de uso. El conjunto de casos de uso deben de tener la posibilidad de representar los requerimientos funcionales y para ello se requiere de una revisión a detalle para esto.

Pasos
- Elaborar casos de uso
- Llenar plantilla de casos de uso
- Someter la plantilla a revisión
- Elaborar casos de uso al escenario principal y sus extensiones

Técnicas que utiliza
- Plantilla de caso de uso
- Casos de uso
- Equivalencia de clases
- Valor límite
- Matriz de dominio

Prueba de control de flujo

Descripción
Identifica los caminos de ejecución a través de un módulo del programa para crear y ejecutar casos de prueba para cubrir esos caminos. Un camino es una secuencia de sentencias de ejecución que comienzan con una entrada y terminan con una salida. Aunque una exhaustiva prueba de los controles de flujo existentes es difícil de llevar a cabo, se debe resaltar su utilidad.

Pasos
- Representar de manera gráfica a los módulos de código
- Identificar el nivel de cobertura
- Realizar la prueba estructurada

Técnicas que utiliza
- Gráfica de control de flujo
- Niveles de cobertura
- Complejidad ciclomática
- Camino base
- Casos de prueba

Prueba estructurada

Descripción
Es conocida como la base de la prueba de trayectoria. Se basa en los trabajos realizados por McCabe. Utiliza un análisis de topología de los gráficos de control de flujo para identificar los casos de uso.

Pasos
- Obtener las gráficas de control de flujo de los módulos de software
- Realizar el cálculo de complejidad ciclomática de las gráficas (C)
- Seleccionar un conjunto de C trayectoria base
- Elaborar un caso de prueba para cada trayectoria base
- Ejecutar las pruebas

Técnicas que utiliza
- Complejidad ciclomática
- Camino base
- Casos de prueba

Prueba de flujo de datos

Descripción
Es una herramienta que se utiliza para detectar el uso inapropiado de valores en los datos por errores de código.

Las variables que contienen valores tienen un ciclo de vida definido. Se crean, se utilizan, y se destruyen. En algunos lenguajes de programación (FORTRAN y BASIC, por ejemplo) la creación y la destrucción son automáticas. Se crea una variable la primera vez que se le asigna un valor y se destruye cuando el programa termina. En otros idiomas (como C, C++ y Java) la creación es formal. Estas declaraciones ocurren generalmente dentro de un bloque del código que comienza con un corchete de abertura "**{**" y terminan con un corchete de cierre "**}**". Las variables definidas dentro de un bloque se crean cuando sus definiciones se ejecutan y se destruyen automáticamente cuando finaliza el bloque. Esto se llama el "alcance" de la variable.

Existen tres posibilidades para la primera ocurrencia de una variable a través de una trayectoria del programa:

1. ~d La variable no existe (indicado por el ~), entonces es definida (d)

2. ~u La variable no existe, entonces es usada (u)
3. ~k La variable no existe, entonces es destruida (k)

La primera que se presenta en la lista es la correcta. La variable no existe y entonces se define. El segundo es incorrecto. Una variable no debe ser utilizada antes de que se defina. El tercero es probablemente incorrecto. Destruir una variable antes de que sea creada es indicativo de un error de programación

Ahora considere los siguientes pares de secuencias de definición (d), uso (u) y destrucción (k):

dd Definida y definida otra vez: no es incorrecto, pero si sospechoso. Probablemente un error de programación.
du Definida y usada: es perfectamente correcto, un caso normal.
dk Definida y luego destruida: no es incorrecto, pero probablemente es un error de programación.
ud Usada y definida: aceptable.
uu Usada y usada otra vez: aceptable.
uk Usada y destruida: aceptable.
kd Destruida y definida: aceptable. Si una variable es destruida y redefinida.
ku Destruida y usada: Un defecto serio. Usar una variable que no existe o es indefinida es siempre un error.
kk Destruida y destruida: Probablemente un error de programación.

Pasos

- Realizar la prueba de control de lujo
- Identificar a las variables de cada módulo
- Elaborar la gráfica de flujo de datos
- Realizar inspección estática de la gráfica de flujo de datos
- Realizar la inspección dinámica del flujo de datos

Técnicas que utiliza

- Gráfica de control de flujo
- Complejidad ciclomática
- Camino base
- Gráfica de flujo de datos
- Inspección estática de la gráfica de flujo de datos
- Inspección dinámica del flujo de datos
- Casos de prueba

Una descripción de las técnicas que se acaban de mencionar en cada una de las pruebas se puede consultar en el Anexo de este documento.

Las pruebas que se han presentado representan unas cuantas de las que existen para el desarrollo de software, lo que no quiere decir que sean las únicas que existen ni tampoco quiere decir que se deban de mantenerse bajo una estructura rígida. La presentación debe de considerarse como un primer acercamiento a estas opciones ya que dependerá de las circunstancias para hacer una selección y uso de estos recursos. En la bibliografía que contiene este documento se podrán encontrar más opciones, que pese a encontrarse en diferentes clasificaciones no se excluye la posibilidad de que sean aplicadas.

El siguiente cuadro se puede ubicar la ejecución de las pruebas de software dentro de las fases del proceso de Desarrollo y mantenimiento (Fig. 6):

Fase / Prueba	inicio	Requerimientos	Análisis y diseño	Construcción	Integración y pruebas	Cierre
Unitarias				X		
Integración					X	
Sistema					X	

Fig. 6 Relación del tipo de pruebas y el proceso de Desarrollo y mantenimiento de software

Cada tipo de prueba agrupa a pruebas de software que se encargan de una parte en específico del software. Esta división no es restrictiva ya que para cada tipo de prueba se pueden encontrar pruebas que apliquen a más de un tipo (Fig. 7).

Tipo / Prueba	Unitaria	Integración	Sistema
Clases de equivalencia	X	X	X
Valor límite	X	X	X
Tabla de decisión	X	X	X
En parejas	X	X	X
Transición de estados			X
Análisis de dominio	X	X	X
Casos de uso			X
Control de flujo	X		
Estructurada	X		
Flujo de estados	X		

Fig. 7 Relación del tipo de pruebas y las pruebas de software

Una prueba se constituye, entre otras cosas, de técnicas y herramientas. En el siguiente cuadro (Fig. 8) se encuentra le relación de cada prueba con las técnicas que utilizan. En el caso de las herramientas se puede encontrar una gran variedad, para este documento sólo se presenta un conjunto de herramientas comunes que no tienen que ver con productos o marcas que puedan encontrarse en el mercado. En el Anexo de este documento puede encontrarse una descripción de las técnicas y herramientas.

Prueba / Técnica	Clases de equivalencia	Valor límite	Tabla de decisión	En parejas	Transición de estados	Análisis de dominio	Casos de uso	Control de flujo	Estructurada	Flujo de estados
Casos de prueba	X	X	X	X	X	X	X	X	X	X
Clases de equivalencia	X	X	X			X	X			
Valor límite		X	X			X	X			
Tabla de decisiones			X							
Arreglos otogonales				X						
Algoritmo allpairs				X						
Diagrama de transición de estados					X					
Tabla de transición de estados					X					
Casos de prueba de transición de estados					X					
Análisis de dominio						X				
Matriz de dominio						X	X			
Plantilla para caso de uso							X			
Gráfica de control de flujo								X		X
Nivel de cobertura								X		
Complejidad ciclomática								X	X	X
Camino base								X	X	X
Gráfica de flujo de datos										X
Inspección estática de la gráfica de flujo de datos										X
Inspección dinámica del flujo de datos										X

Fig. 8 Relación de las técnicas y las pruebas de software

Objetivo 3

Una de las preocupaciones importantes tiene que ver con el control sobre las actividades en el desarrollo, no es difícil de imaginar si se tiene en cuenta el orden tan peculiar con el que se venía trabajando. Antes de utilizar cualquier modelo de análisis complejo se debe comenzar por recolectar datos básicos sobre las actividades. La duración de las actividades resulta un buen principio para comenzar con el análisis. Las métricas funcionan como una herramienta de análisis ya que con ellas pueden manejarse los datos generados por las actividades.

La intención de manejar métricas en la jefatura es que éstas proporcionen el apoyo a la administración de los proyectos internos. De esta manera se cubre el tercer objetivo que fue planteado al inicio: Proporcionar una herramienta cuantitativa para los proyectos de desarrollo de software.

Estas herramientas principalmente se enfocarán en el estado y la duración de los proyectos. Esto permitirá tanto a los líderes de cada proyecto como al encargado de la jefatura tomar las decisiones que consideren pertinentes al considerar los tiempos planificados contra los reales y con ello establecer las acciones correspondientes.

Debido a que no se cuenta con registros históricos sobre la duración de cada una de las actividades que se realizan no se puede establecer una base para realizar estimaciones. Aprovechando la experiencia del equipo se intenta afianzar las estimaciones que se realizan con base a la experiencia de cada uno y confrontarlos a los datos reales que son recolectados.

Con base en el método GQM se realiza la selección de métricas más adecuadas.

Fase de planificación

Se conforma el equipo de GQM con integrantes de la jefatura que no intervienen a profundidad en los proyectos de la jefatura, pero conocen el funcionamiento de éstos. Permite que de esta forma se mantenga la imparcialidad al momento de ejecutar las actividades de medición.

Se establece el área de mejora. Por la definición del proyecto el área a tratar es el relacionado con la administración de los proyectos que integran a la jefatura, ya que es una de las causas que provocan problemas.

Se elabora un plan del proyecto el cual es la base de programa de medición, el cual tiene el siguiente contenido:
Resumen ejecutivo
Introducción
Caracterización
Programación
Organización
Proceso administrativo
Entrenamiento y promoción

Fase de definición

Meta: Tiempo

Analiza la: Duración de cada una de las actividades de los proyectos
Con el propósito de: Comprender la inversión de tiempo de cada una de las actividades más importantes
Con respecto a: - Los tiempos planificados
- Los tiempos reales
Desde el punto de vista de: El administrador de proyectos
Bajo el siguiente contexto: Todos los proyectos de la jefatura

Los objetivos bajo el anterior formato permiten establecer claridad, pues se establecen las intenciones de dichos objetivos y se evitan ambigüedades cuando sólo se enuncia el nombre del objetivo.

Preguntas

Las siguientes preguntas son inquietudes manifestadas por la organización sobre un tema en específico que serán contestadas por los resultados de las métricas. De esta forma se justifica la existencia de cada métrica.

¿Cuál es la actividad que requiere una mayor inversión de tiempo?
¿Cuántas son las actividades que presentan un desfase con respecto a lo planificado?
¿Cuál es la actividad que contiene el mayor desfase?
¿Cuál es el promedio de duración de cada una de las actividades?
¿Cuál es el progreso del proyecto con respecto a lo planificado?
¿Cuál es el avance de cada proyecto con respecto a lo planificado?
¿Qué actividades representan más desfases?
¿Cuál es el tiempo requerido para realizar una prueba?

La escala base de las métricas será en minutos y a partir de ella se realizarán las conversiones en caso de ser requerida por alguna métrica.

Las métricas definidas son revisadas por el equipo GQM con el objetivo de encontrar defectos y realizarle mejoras.

Las fuentes de información para las métricas se obtendrán de los formatos establecidos.

La obtención de datos se realiza conforme al procedimiento definido en el plan de medición antes de que se comience a almacenar los datos, estos se verifican y validan. De acuerdo con el plan de medición se realizan las siguientes actividades de revisión:

- Exactitud de los datos
- Si los datos se encuentran dentro del rango
- Correcta clasificación
- Que los datos se encuentren completos
- Puntualidad de los datos

Recolección y almacenamiento

El almacenamiento y análisis de los datos se realizará en una hoja de cálculo, ya que proporciona flexibilidad y no requiere de más recursos para la jefatura. La periodicidad en la que serán recolectados los datos es semanal con el fin de procesar y entregar los reportes correspondientes.

Análisis y presentación

Para el manejo de los datos se ha dividido los datos en tres secciones: una para datos operativos, otra para el procesamiento de los datos y una final que está destinada para la presentación de los resultados. Éstas últimas contienen las preguntas establecidas en el plan GQM y que se pretenden contestar. La presentación de los resultados se realizará por medio de reportes.

Estrategia de implantación

Para llevar a cabo la implantación de esta propuesta se han considerado algunos elementos básicos. Estos elementos han sido considerados conforme al tamaño y recursos con los que cuenta la jefatura en la que se realizará la implantación.

A continuación se presentan las actividades para llevar a cabo la implantación de la propuesta:
Presentación. La propuesta es presentada a todo el equipo de manera general y se recolectan las opiniones al respecto.
Ajustes. Se realizan ajustes a la propuesta como resultado de la retroalimentación con el equipo tras haber iniciado la operación.
Selección. Se llevará a cabo una selección del proyecto y los integrantes que participarán en la prueba piloto de la propuesta.
Seguimiento y retroalimentación. Durante toda la implementación se tendrá un acercamiento con los equipos para observar su progreso y despejar dudas.
Capacitación. Se organizarán capacitaciones a todos los involucrados de la operación con respecto a la propuesta.
Evaluación. Se realizará una evaluación después de algún un tiempo de utilizar la propuesta para observar la adopción que se ha tenido de la propuesta.
Resultados. Se establecerán lapsos de tiempo en los que se presentará los resultados de la implementación con la finalidad de establecer las siguientes acciones que se deberán de realizar.

Para revisar el funcionamiento inicial de la propuesta sin realizar afectaciones considerables es importante destacar la implantación gradual de la propuesta en la jefatura. Para lograrlo, la opción adoptada es iniciar la operación en un proyecto nuevo o en uno que se encuentre en marcha, pero cualquiera de ellos deberá de considerarse de bajo impacto. De esta forma será posible realizar los ajustes convenientes y generar la experiencia necesaria en el funcionamiento de la propuesta.

Consideraciones para seleccionar al equipo que participará en la implantación de la propuesta:

- Tipo de proyecto. Para la implementación inicial de la propuesta deberá de realizarse en un proyecto que no sea crítico para la empresa.
- Equipo. La disposición del equipo de trabajo que utilice la propuesta deberá contar con el compromiso para realizar las actividades señaladas y realizar aportes que considere necesarios para mejorar la propuesta.
- Nivel de conocimientos. Todos los involucrados deberán de contar con los conocimientos indispensables de la propuesta para involucrarse de manera directa en los temas relacionados.

- Entorno. Es importe que se genere un ambiente laboral propicio para que sea utilizada adecuadamente la propuesta y sean desechados los mitos y confusiones que suelen surgir entorno a una propuesta que modifica la forma en la que la gente realiza su trabajo.
- Adaptación. Antes de intentar avanzar al siguiente nivel de los procesos como lo muestra MoProSoft debe por lo menos encontrarse en operación durante tres proyectos diferentes para realizar los ajustes necesarios.

Capítulo IV Conclusiones

Como parte final se presentan las experiencias, inquietudes, aportaciones y posibles trabajos futuros que surgen a raíz de la elaboración de este proyecto.

Es importante destacar la importancia de la calidad del software actualmente por la relación que tiene el software en muchas de las actividades que realiza el ser humano, por lo que desarrollar software de calidad debe de ser reconocida por los involucrados en su desarrollo y por lo tanto también para la informática.

Todo el proyecto busca de principio mejorar el desarrollo de software de una jefatura para proporcionar un mejor servicio dentro de la empresa, ya que al ser un área de desarrollo interna no cuenta con más clientes que el personal de la empresa.

En este proyecto se han mostrado algunas de las opciones que existen en la industria del software para mejorar la calidad del software y que han sido seleccionadas para beneficio de una empresa. Debido a que ningún modelo consultado llenaba por completo las necesidades trazadas fue necesario elabora la propuesta con recursos que se complementan entre sí. Es por esto que la postura adoptada para este proyecto fue la de atenerse a las circunstancias y adecuar las mejores prácticas de la manera más conveniente para la organización y evitando adecuar la organización al modelo de referencia lo más que fuera posible. De esta forma se logra flexibilidad al momento de aplicar un modelo y también al momento de seleccionar las herramientas de apoyo, ya que no se depende completamente de los productos proporcionados por los modelos.

Con este proyecto, pese a la restricción de recursos en las que se encontraba, se demuestra que no son necesarias grandes inversiones para generar una propuesta que mejore la calidad del software.

El modelo MoProSoft figura como una opción asequible para las organizaciones pequeñas que buscan mejorar y resulta además una ventaja por su compatibilidad con los modelos CMM e ISO 9001.

La selección de elementos para esta propuesta parte de un planteamiento de cuestiones inmediatas y alcanzables que en un momento determinado lograron acordarse y ponerse de manifiesto. Estas cuestiones forman parte de la estrategia de la empresa, logrando de esta forma que la propuesta sea de importancia para ella porque contribuye al logro de la estrategia.

De manera general se puede resumir los puntos relevantes de este documento de la siguiente manera:

- Se plantean los problemas y las características que tiene la organización.
- Se delimitan los problemas englobándolos en tres objetivos que son de importancia por su participación en la estrategia de la empresa.
- Se realiza una investigación para obtener las mejores prácticas dentro de la industria del software.
- La elaboración de una propuesta con las mejores prácticas ajustándose a las características específicas de la organización.
- Las experiencias e inquietudes más notables que se generaron durante el proyecto son presentadas.

Con el proyecto se encontraron los diferentes temas de interés relacionados con la calidad en el software como:

- El problema de la calidad, como idea general, no es en gran medida un problema técnico a comparación de la construcción de caminos para llegar a ella.
- Existe una gran dificultad para elaborar un modelo general que nos garantice la calidad.

- La discrepancia entre modelos o la imposibilidad de elaborar uno que abarque todos hace posible la existencia de varias perspectivas desde las que se puede abordar la calidad del software.
- Para elaborar software con calidad influyen varios factores de los cuales la gente es uno muy importante, porque de ellos depende la ejecución de cualquier propuesta.
- Una de las dificultades para elaborar un modelo de calidad se debe al comportamiento constante de sus elementos que hace posible que los modelos se apliquen en los casos descritos, algo difícil de lograr por la calidad debido a la constante variación en las que se encuentran las apreciaciones sobre este concepto y sus objetos de estudio.
- No existe un criterio establecido y confiable que indique se están realizado las actividades suficientes para alcanzar la calidad.
- Existen muchos criterios de calidad pero no todos tienen el mismo significado para cada organización.
- La calidad no es fija, sino que varía de acuerdo a las circunstancias y es por ello que se tienen varios puntos de partida. La calidad es un punto de apreciación que varía constantemente. No existe un punto de vista privilegiado que nos permita por el momento abarcar este plexo de apreciaciones.

También se encontraron diferentes obstáculos durante en proyecto, como los siguientes:

- Los tiempos asignados a este tipo de proyectos no es el suficiente para realizar un estudio pormenorizado de las diferentes prácticas que existen en la industria.
- Los modelos más reconocidos tienen una estructura poco descriptiva lo que dificulta su interpretación. Aunado a esto cabe decir que puede resultar difícil llevar a cabo las sugerencias del modelo en organizaciones pequeñas.
- La definición de objetivos puede llegar a ser problemático, pues muchas veces no son fáciles de establecer, además de que pueden encontrarse intereses creados que relegan a los objetivos que son necesarios.
- Otra dificultad tiene que ver con el aspecto humano durante todo el proceso, porque se requiere de un esfuerzo considerable, si es que no se cuenta con la habilidad para hacerlo, para unificar la cultura laboral de un grupo para que realmente se obtengan los resultados esperados.
- Generar credibilidad después de algunos intentos anteriores no muy afortunados resulta difícil en este tipo de proyectos en donde las ventajas no son evidentes al inicio.
- Entender a la organización cuando la gente no está muy dispuesta a participar en el proyecto o cuando su forma de trabajo aún se mantiene ambigua.
- Tratar de establecer una línea base se vuelve una tarea laboriosa cuando el ambiente en el que opera la empresa cambia constantemente y siempre se busca ingresar nuevos elementos a los ya establecidos, provocando ajustes de último momento que afectan a las demás actividades.
- Resulta imposible llevar a cabo todas las buenas prácticas que existen actualmente por varias razones como: las necesidades de la empresa, los enfoques de cada propuesta, el presupuesto con el que se dispone, el tiempo con el que se cuenta, etc.
- Otro problema no es la ausencia de material de referencia, sino al contrario, su amplitud y la falta de un criterio infalible para seleccionar aquella que nos garantice el logro de los objetivos.
- Aún no existen un conocimiento objetivo para lograr la calidad, por lo que se depende en gran medida del buen juicio de la persona al mando que esto pueda lograrse.
- No es posible probar todos los casos que existen en los sistemas tan complejos con los que se cuenta actualmente ya que nos encontramos ante una gran cantidad de variables dependientes entre ella, por lo que un sistema libre de fallas resulta imposible de conseguir.
- Pese a que mucho de los puntos de este documento son presentados con claridad y sencillez en realidad no representan el esfuerzo realizado para lograrlo con la colaboración de muchas personas.

Del proyecto también se desprenden aportaciones significativas para diferentes perspectivas.

A partir de las necesidades específicas de una empresa y del estudio realizado, se descubrió que la inquietud no solo aquejaba a la empresa en cuestión sino que es un asunto recurrente en la industria del software en México. Es un acercamiento a los problemas que enfrenta la industria del software y específicamente las organizaciones que se dedican al desarrollo y mantenimiento de software, no solo dentro del país sino también a nivel internacional.

Para la empresa: los procesos empleados le permitirán realizar el trabajo de manera más ordenada y clara. Las métricas proporcionan un medio de análisis de las actividades que realiza. Y las pruebas son las herramientas que permitirán corroborar el buen funcionamiento del software. Que en su conjunto van dirigidos al logro de sus objetivos estratégicos. Los procesos seleccionados han sido adoptados sin muchas variaciones por lo que es posible llevar a cabo una verificación oficial del modelo para obtener el comprobante de haber alcanzado cierto nivel de madurez en dichos procesos.

Para el ambiente académico se proporciona un caso de estudio en el que se aplicar los conocimientos teóricos a una situación real y del cual se desprenden un conjunto de temas a tratar con una bibliografía que sustenta el conocimiento aplicado.

Para el estudiante o persona que se está iniciando en los temas referentes a la calidad de software se presenta un ejemplo en el que se pretende resolver un problema de calidad de software con el conocimiento que se encuentra presente en la industria del software. De esta forma se intenta mostrar uno de los retos que deberán de ser resueltos en el mundo real. Y de ninguna forma se pretende que este proyecto sea tomado como una imposición al que deban de alinearse todos los casos, sino sólo abordarse como una de las posibles soluciones ante este tipo de problemas.

Para el encargado de implementar el modelo MoProSoft encontrará herramientas de apoyo que podrán utilizarse con el modelo y que no se encuentran mencionados en éste.

Si bien es cierto que se identificaron a lo largo del proyecto muchas oportunidades de mejora y se encontraron opciones para solucionarlos no se optó por sobrepasar los requerimientos establecidos en este caso para generar un valor agregado, ya que muchas de ellas requieren de recursos que por mínimos que sean tendrán un impacto sobre los que han sido asignados.

En cuanto a las dificultades relacionadas con el tema de la medición en el software podemos encontrar las siguientes.

Para realizar una medición exitosa se debe tener en cuenta que las características de las entidades candidatas deberán de conservarse iguales en todo momento para que sin importar de quién las realice y bajo un conjunto de reglas bien establecidas el resultado se mantenga siempre igual. En el caso de la calidad y del software las características no se conservan constantes y se ven ampliamente dependiente de las circunstancias en las que se desea realizar una evaluación.

Las mediciones para las características del software son complejas y ambiguas debido en parte a que no tienen un acuerdo general de qué medir y cómo medir.

Una de las cosas que dificultan las mediciones en el software es que no se tiene un profundo entendimiento de sus atributos de las cuales se derivan las sofisticadas herramientas de medición.

Las relaciones empíricas para un atributo no siempre se llegan a un acuerdo, especialmente cuando reflejan preferencias personales.

Al problema de cómo llevar a cabo la medición de ciertos atributos del software le antecede el problema de qué es lo que se pretende medir, el cual se encuentra necesariamente bajo una interpretación definida de calidad a la que se pretende llegar con ello.

No todos los criterios de la calidad cuentan con una métrica. Algunas propiedades no son fáciles de medir sobre una escala numérica.

Es difícil obtener una relación directa de la métrica con lo que se quiere medir en aquello que no es fácil de definir.

Existen pretextos típicos utilizados por la gente para no utilizar métricas:

- Cada proyecto es único
- La tecnología cambia rápidamente
- El resultado del proyecto sólo es el reflejo de las características de las personas que trabajan en el proyecto
- No importan los proyectos futuros, sólo los actuales.

Un plan de métricas que se enfoque en la recolección de datos más que en la definición de un plan claro de cómo utilizar los datos no augura un buen término, pues de poco o nada servirá una gran cantidad de datos procesados si no atienden a las necesidades que imperan en ese momento.

Un plan de medición riguroso de todas las actividades podrá hasta cierto grado proporcionar una constante en actividades que son repetitivas, pero no puede realizarse de la misma manera con aquellas actividades creativas o de innovación y mucho menos garantizar un pronóstico en la duración de dichas actividades que nunca se han realizado.

La falta de formalidad y consensos para realizar las mediciones en el software es lo que ha creado un ambiente de escepticismo en la implementación de los programas de medición.

Es un error considerar que la información creada por las métricas para generar pronósticos deba de ser necesariamente exacta, pues la experiencia generada es insuficiente para garantizar que en la realidad las cosas ocurran como se plantea. La información tiene que considerarse como puntos de referencia de la que deberán de tener sus reservas.

Este proyecto representa también un punto de partida en la que se descubren inquietudes que requieren de un estudio más profundo. Entre las inquietudes más notorias podemos encontrar las siguientes:

- ¿Cuál es la estructura que tiene la calidad?
- ¿Es correcto hablar de sólo un concepto de calidad o de la existencia de varios conceptos de calidad?
- ¿Cuál es la relación entre calidad, necesidad y utilidad?
- ¿Cuál es el papel de la ética dentro de la calidad?
- ¿Cuáles son los elementos que hacen posible que un concepto de calidad sea adoptado en un momento determinado?
- ¿Cuál será la forma para eliminar la brecha el modelo conceptual y la práctica para que sea posible concretar los beneficios enunciados por el modelo?
- ¿De qué manera se construye un juicio lo suficientemente confiable para seleccionar los elementos del conocimiento de se emplearán en un momento determinado?

Existen cuestiones que quedan pendientes porque rebasan los alcances planteados por este proyecto, pero resulta importante mencionarlos como parte de la continuidad del proyecto.

- Evaluación posterior de la organización en los procesos implementados mediante la herramienta EvalProSoft.
- Evaluación de los objetivos planteados en el proyecto para determinar si fueron alcanzados.

- Continuar con la implementación del siguiente nivel de los procesos adoptados.
- Seleccionar las herramientas de apoyo que permitan automatizar algunas de las actividades.
- Realizar reuniones programadas para darle continuidad al plan de calidad y realizar los ajustes que se consideren convenientes.
- Elaborar material y cursos de capacitación conforme se vayan identificando las mejores prácticas que se vayan generando en la ejecución con el fin de transmitir el conocimiento.

Anexos

Roles y proyectos de la jefatura

Líder de proyecto	Proyecto	No. De dependencias	Grado de dependencia	Ambiente	No. De desarrolladores	Documentación	Ambiente de pruebas	Tipo de aplicación	Lenguaje de programación	Base de datos
1	1	2	Alto	Producción	2	Sí	Sí	OLAP	C#	Oracle
	2	1	Alto	Producción	0	Sí	No	Cli. – Serv.	C#	Oracle
	3	1	Alto	Producción	2	Sí	No	OLAP	C#	Oracle
	4	4	Alto	Desarrollo	2	Sí	Sí	OLAP	C#	Oracle
2	5	1	Alto	Producción	2	Sí	Sí	Reportes	Java	Oracle, SQL Server
	6	4	Alto	Desarrollo	2	Sí	Sí	Reportes	Java	Oracle, SQL server
	7	4	Alto	Desarrollo	4	Sí	No	Data warehouse	Pendiente	Oracle
3	8	4	Alto	Desarrollo	1	Sí	Sí	Web	.net	Oracle
	9	0	Alto	Desarrollo	Pendiente	Sí	Sí	Cubo	SQL	Oracle
	10	0	Alto	Producción	1	Sí	No	Business intelligence	Software de terceros	Oracle
	11	0	Alto	Producción	Pendiente	Sí	Sí	Cli. – Serv.	ASP	Oracle
	12	1	Alto	Desarrollo	Pendiente	Sí	Sí	Web	C#	Oracle
	13	2	Alto	Producción	Pendiente	Sí	Sí	Web	Software de terceros	Oracle
	14	2	Alto	Producción	Pendiente	Sí	No	Cli. – Serv.	VB	Oracle
	15	Pendiente	Pendiente	Desarrollo	Pendiente	Sí	No	Pendiente	Pendiente	Oracle

Roles definidos en la jefatura

- Líder de equipo
- Líder de proyecto
- Líder funcional
- Desarrollador
- Encargado de pruebas
- Auditor

Métodos y técnicas

1. *Work Breakdown Structure*

El *Work Breakdown Structure* (WBS) es una descomposición jerárquica en forma de árbol (que va de lo general a lo específico) del trabajo que necesitan ser realizado por el equipo del proyecto para lograr el objetivo del proyecto y crear los entregables requeridos. El WBS organiza y define el alcance total del proyecto. El WBS divide el proyecto en unidades más pequeñas de trabajo, las cuales son más administrables y detalladas conforme se desciende de nivel. En el último nivel se encuentran los componentes llamados "paquetes de trabajo". El nivel de detalle para los paquetes de trabajo variará con el tamaño y la complejidad del proyecto.

2. Acercamiento de arriba hacia abajo (descomposición)

Se comienza con el objetivo del proyecto y sucesivamente se van haciendo divisiones del trabajo hacia abajo a los niveles inferiores hasta que los participantes consideran que el trabajo ha sido lo suficientemente definido. La definición debe ser lo suficientemente detallada que permita la estimación de costos, tiempo y recursos requeridos.

3. Acercamiento de abajo hacia arriba

Es un acercamiento más parecido a una lluvia de ideas. Partiendo del objetivo del proyecto, el equipo de planificación es dividido en grupos para atender cada una de las actividades del primer nivel. Cada grupo identifica las actividades que considera necesarias para lograr las actividades del primer nivel. Una vez logrado, cada grupo lo presenta a los demás para obtener su aprobación, encontrar omisiones y eliminar redundancias con respecto a los otros grupos.

4. WBS para proyectos pequeños

Para proyectos pequeños (en los que pueden ir de uno a tres integrantes en el equipo) es posible utilizar la técnica de mapas mentales. Es una representación gráfica no secuencial de los registros mentales acerca de un tema en específico. Que para este caso es en torno al objetivo del proyecto que se le relacionan todos los elementos necesarios para alcanzarlo.

5. Plantillas WBS

Aunque cada proyecto es único, el WBS de un proyecto previo puede con frecuencia ser utilizado como un ejemplo para un nuevo proyecto, ya que muchos proyectos cuentan con un ciclo de proyecto similar y por lo tanto algunos entregables requeridos en cada fase.

6. Método de diagramación precedente (*precedece diagramming method*)

Utiliza cajas o rectángulos, referidos como nodos, para representar actividades y conexiones con flechas que representan las relaciones.

Incluye cuatro tipos de dependencias o relaciones de precedencia.

- Fin – Inicio. El inicio de una actividad depende de que se haya completado la actividad predecesora.
- Fin – Fin. Para que una actividad sucesora sea completada depende de que sea completada la actividad predecesora.
- Inicio – Inicio. El comienzo de una actividad sucesora depende del inicio de la actividad predecesora.
- Inicio – Fin. Para que una actividad sucesora sea completada depende del comienzo de la actividad predecesora.

El tipo de dependencia que describe la relación entre actividades es determinado como resultado de restricciones que existen entre actividades. Cada tipo de restricciones es factor para seleccionar una de las cuatro formas de dependencia. Existen cuatro tipos de restricciones que afectarán la secuencia de las actividades del proyecto y por lo tanto la dependencia entre las actividades.

- Restricciones técnicas
- Restricciones administrativas
- Restricciones entre proyectos
- Restricciones de fechas

7. Método de diagramación por flechas (*arrow diagramming method*)

Utiliza flechas para representar a las actividades y conectarlos a nodos para mostrar sus dependencias. Sólo hace uso de la dependencia Fin – Inicio, por lo que puede requerir del uso de actividades simuladas, que son representadas por líneas segmentadas, para definir todas las relaciones correctamente.

8. Plantillas de red de programación

Pueden ser utilizadas para apresurar la preparación de las redes de actividades programadas del proyecto. Pueden incluir un proyecto completo o tan sólo una parte de ello. Las subredes son útiles cuando el proyecto incluye entregables idénticos o parecidos.

9. Aplicando adelantos y retrasos

Determina las dependencias que podrían requerir un adelanto o un atraso entre actividades para ajustar la relación lógica.

Un adelanto permite una anticipación de la actividad sucesora.
Un retraso pospone a la actividad sucesora.

10. Similar a otra actividad

Algunas actividades del WBS pueden ser similares a actividades ya completadas en otros proyectos. La recolección de esa información puede utilizarse en la estimación de actividades actuales.

11. Estudio de datos históricos

Una metodología de administración de proyectos contiene registros de la duración de las actividades reales y estimadas. Dichos datos históricos pueden ser utilizados en otros proyectos. Los datos registrados serán la base de conocimiento para estimar la duración de las actividades. Esta técnica usa registros históricos.

12. Consejo experto

Se emplea en proyectos innovadores en los que no se cuenta con la experiencia necesaria dentro de la organización. En dicho caso será necesario buscar asesoría externa.

13. Técnica Delphi

La técnica Delphi puede proporcionar una buena estimación en caso de carecer del consejo de un experto. Este es un grupo técnico que extraen y resumen el conocimiento del grupo para lograr una estimación. El grupo es introducido al proyecto y en la naturaleza de la actividad, a cada individuo del grupo es cuestionado para que haga su mejor conjetura en la duración de la actividad. Los resultados son tabulados y presentados al grupo como histograma con la etiqueta de primera pasada. A los participantes en los que han caído fuera de los cuartiles se les pregunta las razones de su conjetura para que sea compartida a los demás. Después de escuchar las razones se les vuelve a pedir a todos los miembros a que realicen nuevamente su conjetura. Los resultados son presentados en el histograma con la etiqueta de segunda pasada y nuevamente las conjeturas fuera de los cuartiles son argumentadas. Una tercera conjetura es realizada con su respectivo histograma y etiquetado como tercera pasada.

14. Técnica de los tres puntos (*three-point estimates*)

La exactitud de la duración de la actividad puede ser mejorada considerando el riego en la estimación original. La estimación es basada en tres tipos de estimaciones:

- Más probable. Es el tiempo usualmente experimentado en ese tipo de actividades considerando los posibles recursos asignados, la productividad, expectativas reales, dependencias de los participantes e interrupciones.
- Optimista. Se encuentra basado en el mejor escenario de la estimación más probable.
- Pesimista. La duración de la actividad se encuentra basado en el peor escenario de la estimación más probable.

Para esto se está recurriendo a la memoria de los profesionales que han trabajado en actividades similares y de los cuales no se cuenta con registros históricos. Una vez conseguido lo anterior es posible apoyarse de la siguiente fórmula:

$$E = \frac{O + 4M + P}{6}$$

En donde:

E = Estimación
O = Optimista
P = Pesimista
M = Más probable

15. Técnica Delphi de banda ancha

Es como resultado de la combinación Delphi y la técnica de los tres puntos. Incluye un panel como el de la técnica Delphi. En lugar de una sola estimación a lo miembros se les solicita que en cada iteración proporcionen su conjetura optimista, pesimista y más probable para la duración de la actividad seleccionada. Los resultados serán recolectados y cualquier estimación extrema será removida. Los porcentajes serán computados para cada una de las tres estimaciones y los porcentajes son utilizados como estimaciones optimista, pesimista y más probable en la duración de la actividad.

16. Consejo experto

Es requerido para situaciones en las que necesita decidir los recursos pero no se cuenta con ninguna experiencia. Cualquier grupo o persona con conocimiento especializado en planificación y estimación de recursos pueden proveer esta experiencia.

17. Estimación de abajo hacia arriba

La programación de actividades no puede ser estimada sin un razonable grado de confianza, es por ello que el trabajo en la programación de actividades es descompuesto de manera más detallada. Los recursos requeridos en los pedazos de trabajo más detallados son recolectados hasta contar con los recursos requeridos. La programación de las actividades podría o no tener dependencias entre ellos que afecten la aplicación y uso de recursos.

18. Estimación de abajo hacia arriba

Esta técnica involucra la estimación de costos de los paquetes individuales de trabajo o programación individual de trabajo con el nivel de detalle más bajo. Este costo detallado es recolectado hasta los niveles más altos

19. Estimación paramétrica

Es una técnica que utiliza una relación estadística entre datos históricos y otras variables para calcular una estimación de costos.

20. Análisis de programación de red (*Schedule network analysis*)

Emplea un modelo de programación y varias técnicas de análisis, como son camino crítico, cadena crítica, análisis de qué pasaría si y nivelación de recursos para calcular las fechas y programar el inicio y fin de las actividades

21. Ruta crítica

Es el camino largo o secuencia de actividades (en términos de duración de la actividad) a través del diagrama de red. Conduce la fecha de finalización del proyecto. Cualquier retraso para completar cualquiera de estas actividades en la secuencia retrasará la finalización del proyecto. El administrador del proyecto pone especial interés en las actividades del camino crítico.

22. Revisión de documentos

Una revisión estructurada podría incluir planes, suposiciones, archivos anteriores del proyecto y otra información. La calidad de los planes así como la consistencia entre esos planes con los requerimientos del proyecto y suposiciones podrían indicar los riesgos en el proyecto.

23. Técnicas de obtención de información (*information gathering techniques*)

Entre ellas podemos encontrar:

- Lluvia de ideas
- Técnica Delphi
- Entrevistas
- Identificación de la causa raíz
- Análisis de fortalezas, oportunidades, debilidades y amenazas

24. Análisis de lista de verificación

La lista de identificación de riesgos puede ser elaborada con base en datos históricos y conocimientos que se han acumulado de anteriores proyectos similares y de otras fuentes de información.

25. Probabilidad del riesgo y determinación del impacto

Se investiga la probabilidad de que cada riesgo en específico pueda ocurrir. La determinación del impacto del riesgo investiga el efecto potencial en un proyecto ya sea en tiempo, costo, alcance o calidad, incluyendo los efectos negativos para las amenazar y los efectos positivos para las oportunidades.

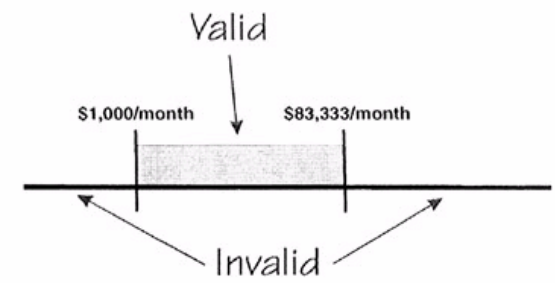
26. Clases de equivalencia

Consiste en generar clases de equivalencia y un valor para cada clase de equivalencia.

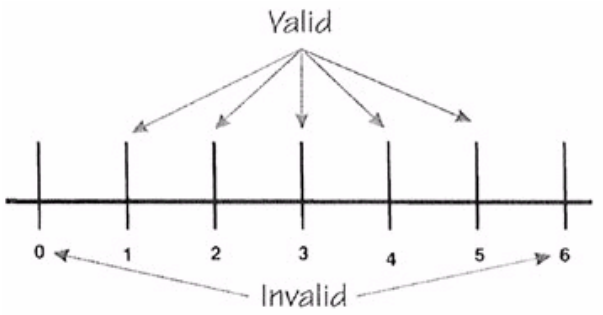
- Una clase de equivalencia consiste en un grupo de valores de entrada en un módulo y son tratados por igual o deberían de producir el mismo resultado.
- De todas las entradas posibles del mismo tipo a un módulo se deberán de detectar aquellos valores que tienen el mismo tratamiento o arrojan el mismo resultado y de éstos se generará una clase de equivalencia.
- Debido a que todos los valores dentro de una clase son tratados de manera uniforme, si una falla es descubierta por un caso de prueba los demás valores que integran la clase también lo estarán. Por lo que no es necesario probar cada uno de los valores de cada clase.
- Una vez que se tengan definidas las clases de equivalencia se selecciona un valor de entrada por cada clase de equivalencia.

Tipos de clases de equivalencia:

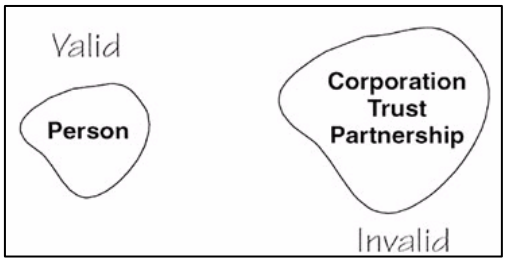
- En los que se puede ingresar cualquier valor dentro de un rango.



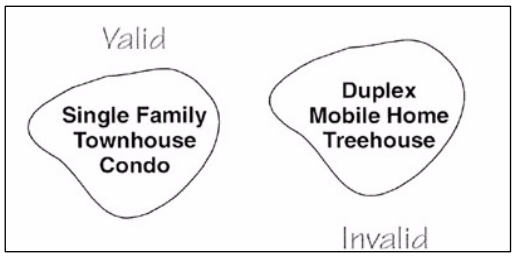
- En los que se puede ingresar alguno de los valores definidos dentro de un rango.



Selección simple de clases equivalentes.



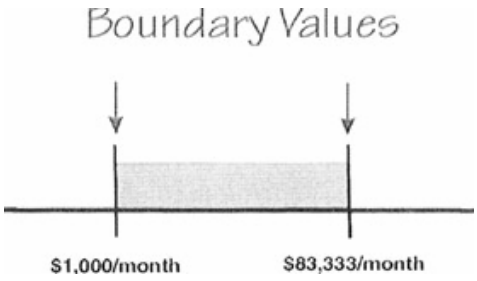
Selección múltiple de clases equivalentes.

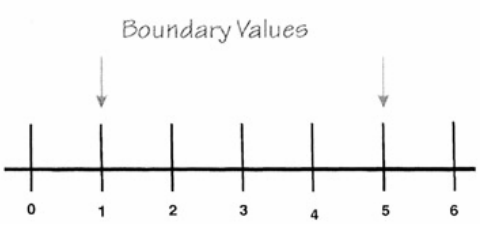


27. Valor límite

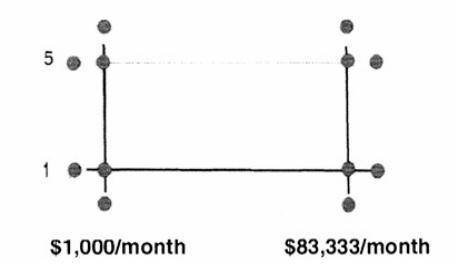
Consiste en la identificación de los valores que son los límites de cada clase de equivalencia.

- Identifique todas las clases de equivalencia con la técnica Clases de equivalencia.
- Para cada clase de equivalencia se deberá de identificar los valores límite.





- Identificar los valores que están por debajo y por encima de cada valor límite.



28. Tabla de decisiones

Es una representación de reglas basadas en condiciones y acciones.

- Representar cada una de las entradas como condición.
- Establecer los valores o rango de valores de cada condición.
- Generar una regla por cada una de las combinaciones de los valores o rango de valores de las condiciones.
- Asignar una acción a cada una de las reglas.
- Llenar la tabla con los valores

	Rule 1	Rule 2	...	Rule p
Conditions				
Condition-1				
Condition-2				
...				
Condition-m				
Actions				
Action-1				
Action-2				
...				
Action-n				

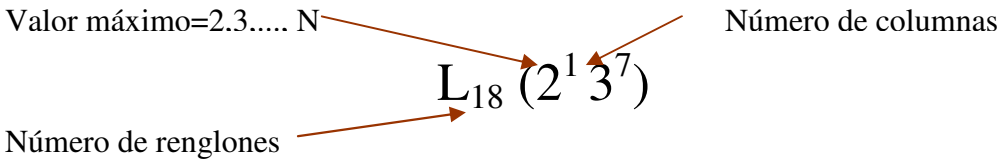
	Rule 1	Rule 2	Rule 3	Rule 4
Conditions				
Condition-1	0–1	1–10	10–100	100–1000
Condition-2	<5	5	6 or 7	>7
Actions				
Action-1	Do X	Do Y	Do X	Do Z
Action-2	Do A	Do B	Do B	Do B

29. Arreglos ortogonales

Un arreglo ortogonal es un arreglo de número en dos dimensiones que tiene su propia característica, seleccionando dos columnas cualesquiera del arreglo y todas las combinaciones con los valores ocurrirán en todas las columnas que sean seleccionadas en parejas. Se puede dar el caso en el que estas combinaciones entre valores de dos columnas se repitan varias veces, pero

ese número de repeticiones se presentarán de igual manera al seleccionar otras dos columnas aunque no sea en el mismo lugar, ya que los arreglos ortogonales son balanceados.

- Identificar todas las variables.
- Identificar el número de valores de cada variable.
- Seleccionar un arreglo ortogonal que tenga el mismo número de columnas que las variables y los niveles de cada columna corresponda a los valores de cada variable. En caso de no encontrar el tamaño exacto seleccione uno de mayor tamaño.
- Colocar las variables y los valores de cada una al arreglo ortogonal.
- Realizar la notación del arreglo.



	1	2	3	4	5	6	7	8
1	1	1	1	1	1	1	1	1
2	1	1	2	2	2	2	2	2
3	1	1	3	3	3	3	3	3
4	1	2	1	1	2	2	3	3
5	1	2	2	2	3	3	1	1
6	1	2	3	3	1	1	2	2
7	1	3	1	2	1	3	2	3
8	1	3	2	3	2	1	3	1
9	1	3	3	1	3	2	1	2
10	2	1	1	3	3	2	2	1
11	2	1	2	1	1	3	3	2
12	2	1	3	2	2	1	1	3
13	2	2	1	2	3	1	3	2
14	2	2	2	3	1	2	1	3
15	2	2	3	1	2	3	2	1
16	2	3	1	3	2	3	1	2
17	2	3	2	1	3	1	2	3
18	2	3	3	2	1	2	3	1

Catálogo de algunos arreglos ortogonales disponibles:

- http://www.research.att.com/~njas/oadir/index.html
- Phadke, Madhav Shridhar. *Quality Engineering Using Robust Design*. New Jersey, Prentice Hall, 1989.

30. Algoritmo de todos los pares (Allpairs)

Es un algoritmo que genera los pares directamente sin emplear recursos adicionales. En el mercado ya se encuentran disponibles herramientas y ejemplos en diferentes lenguajes de programación que generan automáticamente las combinaciones de todos los pares.

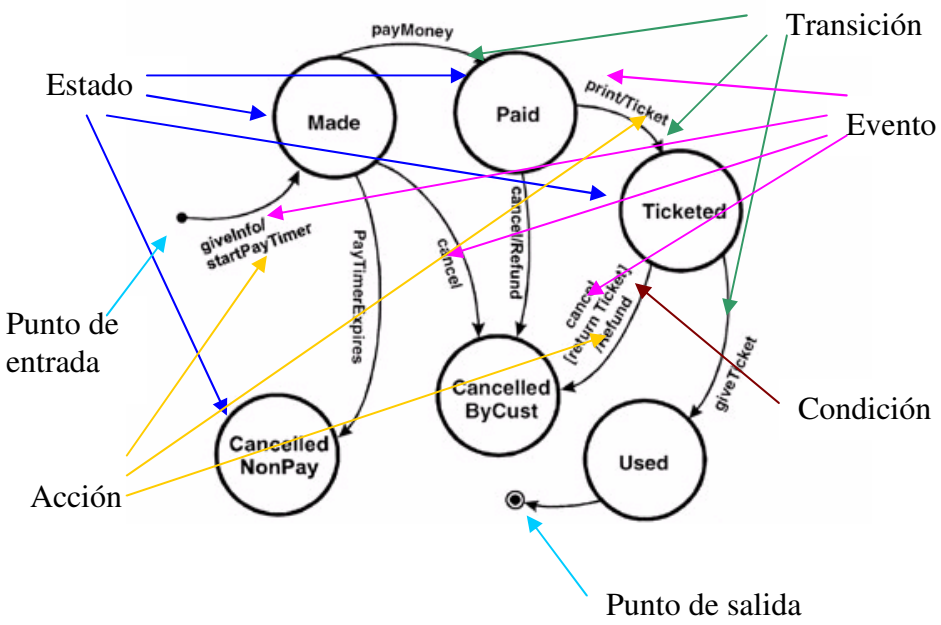
Se puede encontrar un algoritmo de este tipo en la siguiente referencia:

- Kaner, Cem. *Lessons learned in software testing a context-driven*. New York, Wiley, 2002.

31. Diagrama de transición de estados

Es una herramienta que permite realizar una representación gráfica de una entidad específica de un sistema. La notación que se presenta fue creada por Mealy, G.H, aunque existen otras disponibles. El diagrama no muestra todos los posibles estados, eventos y transiciones de una entidad específica.

- Los círculos representan estados. Es una condición en la cual un sistema está en espera por uno o más eventos. Los estados recuerdan las entradas que el sistema ha recibido y define la forma en la que el sistema debería de responder en subsecuentes eventos cuando ocurran. Esos eventos podrían causar transiciones de estados y/o iniciar acciones. El estado es generalmente representado por lo valores de uno o más variables dentro de un sistema.
- Las flechas representan la transición. Representa un cambio entre un estado a otro causado por un evento.
- La descripción entre las flechas es un evento. Es algo que causa que el sistema cambie de estado. Generalmente proviene del mundo y se ingresa al sistema por medio de una interfase. Otras ocasiones son generadas de manera interna por el sistema. Los eventos son considerados por ser instantáneos. Los eventos pueden ser independientes o causalmente relacionados. Cuando un evento ocurre el sistema puede cambiar de estado o permanecer en el mismo estado y/o ejecutar una acción. Los eventos pueden tener parámetros asociados con ellos.
- El comando después de la "/" es una acción. Es una operación iniciada por el cambio de un estado. Con frecuencia estas acciones crean salidas del sistema. Nótese que las acciones ocurren en la transición de estados. Los estados por sí mismos son pasivos.
- El punto negro indica el inicio del diagrama.
- El punto negro con margen indica el fin del diagrama.
- Los corchetes indican un condicional que debe de ser evaluado. Actúa como un filtro que permite la transición sólo si la condición es verdadera.



32. Tabla de transición de estados

Es una forma de representar el comportamiento del sistema de manera completa y sistemática. Consiste en una tabla con cuatro columnas. En esta tabla se pueden enlistar todas las posibles combinaciones de transición de estados.

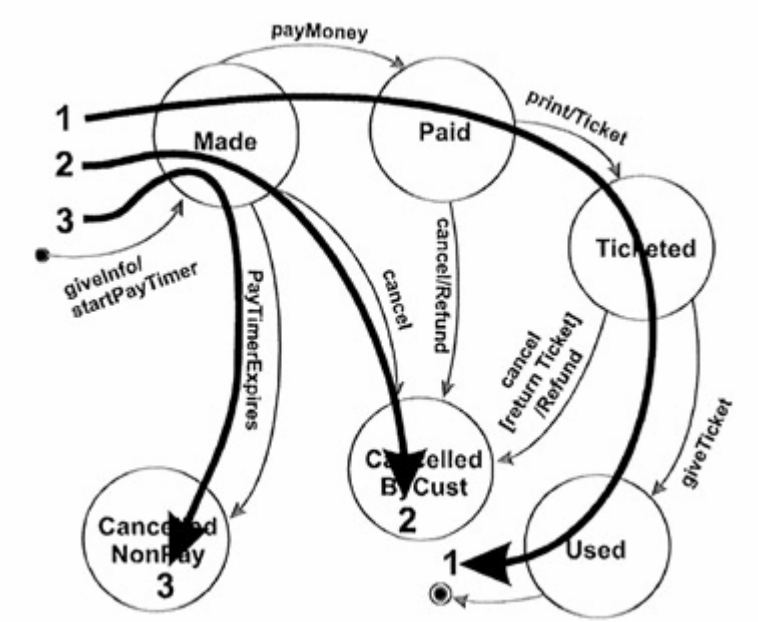
- Identificar los estados
- Identificar los eventos
- Identificar las acciones
- Identificar las posibles combinaciones en la transición de estados
- Colocar las combinaciones en la tabla de transiciones

Current State	Event	Action	Next State
null	giveInfo	startPayTimer	Made
null	payMoney	--	null
null	print	--	null
null	giveTicket	--	null
null	cancel	--	null
null	PayTimerExpires	--	null
Made	giveInfo	--	Made
Made	payMoney	--	Paid
Made	print	--	Made
Made	giveTicket	--	Made
Made	cancel	--	Can-Cust
Made	PayTimerExpires	--	Can-NonPay

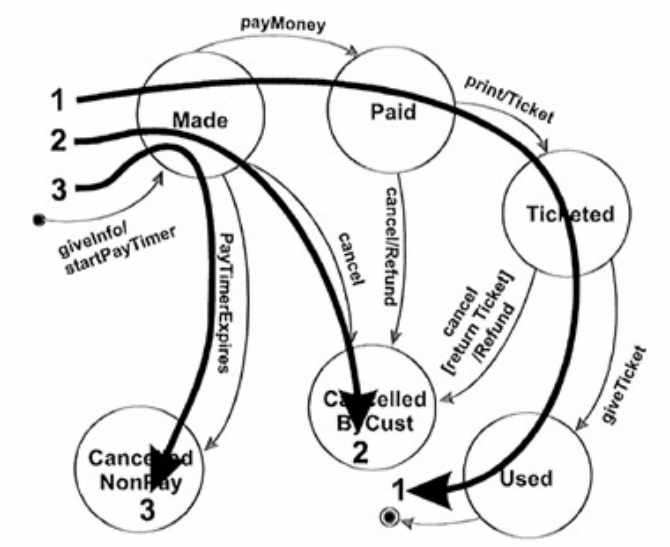
33. Casos de prueba de transición de estados

La información en los diagramas de transición de estado se puede utilizar fácilmente para crear casos de la prueba. Cuatro diversos niveles de cobertura pueden ser definidos:

1. Cree un sistema de casos de la prueba tales que todos los estados "sean recorridos" por lo menos una vez bajo prueba. El sistema de tres casos de la prueba mostrados abajo cumplen este requisito. Éste es generalmente un nivel débil de la cobertura de la prueba.



2. Cree un sistema de casos de prueba tales que todos los acontecimientos están accionados por lo menos una vez bajo prueba. Observe que los casos de la prueba que cubren cada acontecimiento pueden ser iguales que los que cubran cada estado. Una vez más, éste es un nivel débil de la cobertura.



3. Cree un sistema de casos de prueba tales que todas las trayectorias están ejecutadas por lo menos una vez bajo prueba. Mientras que este nivel es preferido más debido a su nivel de la cobertura, puede no ser factible. Si el diagrama de transición de estado tiene lazos, después el número de trayectorias posibles puede ser infinito. Por ejemplo, dado un sistema con dos estados, A y B, donde A es la transición a B y B es la transición a A. Algunas de las trayectorias posibles son:

A→B

A→B→A

A→B→A→B→A→B

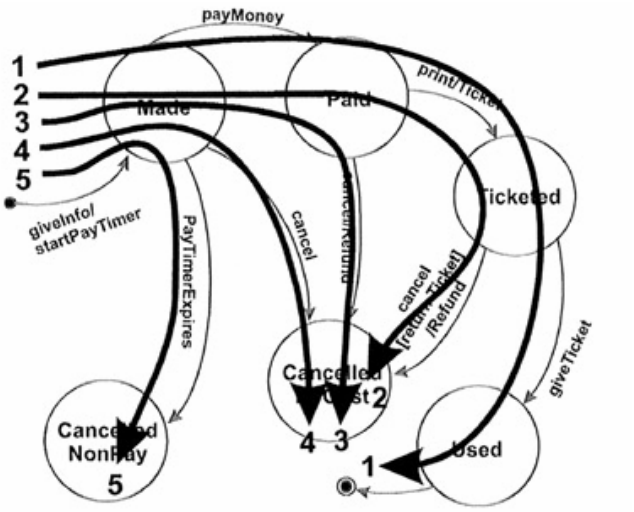
A→B→A→B→A→B→A

A→B→A→B→A→B→A→B→A→B

…

…etcétera por siempre. La prueba de lazos tales como esto puede ser importante si pueden dar lugar a acumular errores computacionales o pérdida de recursos (bloqueos, desperdicio de memoria, etc.).

4. Cree un sistema de casos de la prueba tales que todas las transiciones están ejercitadas por lo menos una vez bajo prueba. Este nivel de la prueba proporciona un buen nivel de la cobertura sin la generación de una gran cantidad de pruebas. Este nivel es generalmente el más recomendado:



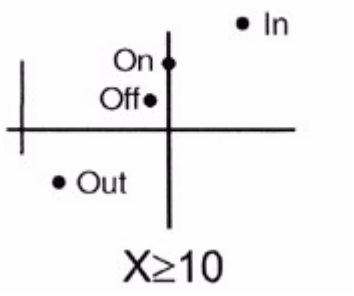
5. Las cajas de la prueba también se pueden leer directamente en la tabla de transición de estado. Las filas grises en la tabla siguiente demuestran todas las transiciones válidas.

Current State	Event	Action	Next State
null	giveInfo	startPayTimer	Made
null	payMoney	--	null
null	print	--	null
null	giveTicket	--	null
null	cancel	--	null
null	PayTimerExpires	--	null
Made	giveInfo	--	Made
Made	payMoney	--	Paid
Made	print	--	Made
Made	giveTicket	--	Made
Made	cancel	--	Can-Cust
Made	PayTimerExpires	--	Can-NonPay
Paid	giveInfo	--	Paid
Paid	payMoney	--	Paid
Paid	print	Ticket	Ticketed
Paid	giveTicket	--	Paid
Paid	cancel	Refund	Can-Cust
Paid	PayTimerExpires	--	Paid

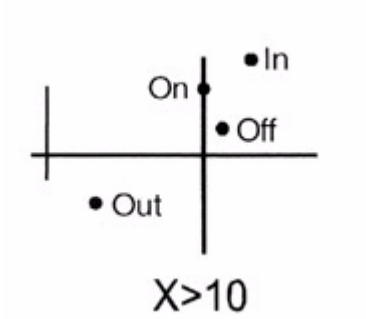
Además, dependiendo del riesgo del sistema, usted puede desear crear los casos de la prueba para alguno o todos los pares inválidos de estado/evento a cerciorarse del sistema no han puesto las trayectorias en ejecución inválidas.

34. Análisis de domino

- Un punto **On** es un valor que se encuentra en el límite.
- Un punto **Off** es un valor que no se encuentra en el límite.
- Un punto **In** es un valor que satisface todas las condiciones del límite pero no se encuentra en el límite.
- Un punto **Out** es un punto que no satisface cualquier condición del límite.
- Un límite se encuentra cerrado cuando se encuentra delimitado por operadores lógicos como ≤, ≥, o =, esos puntos son considerados como dentro del dominio, entonces se coloca un punto **On** se coloca en el límite y es incluido dentro del dominio. Un punto **Off** se encuentra fuera del dominio.



- Un límite se encuentra abierto cuando se encuentra delimitado por los operadores lógicos < o >, esos puntos no son incluidos en el dominio, entonces un punto **On** se encuentra en el límite pero no dentro del dominio. Un punto **Off** se encuentra dentro del dominio.



Robert Binder, sugiere una tabla para documentar los casos de la prueba del análisis de dominio 1x1.

35. Matriz de dominio

- Para cada condición de relación (≤, <, >, ≥) se debe de escoger un punto **On** y un punto **Off**.
- Para condición de igualdad (=) se debe de escoger un punto **On** y dos puntos **Off**, uno ligeramente menor al valor del condicional y otro ligeramente mayor que ese valor.

Variable/ Condition Type			Test Cases															
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
X1	C11	On																
		Off																
	C12	On																
		Off																
	...	On																
		Off																
	C1m	On																
		Off																
	Typical	In																
X2	C21	On																
		Off																
	C22	On																
		Off																
	...	On																
		Off																
	C2m	On																
		Off																
	Typical	In																
Expected Result																		

36. Plantilla para casos de uso

Con la finalidad de proporcionar una representación más detalla a los casos de uso Alistair Cockburn proporcionó un plantilla. Esta plantilla es una adaptación de ese trabajo.

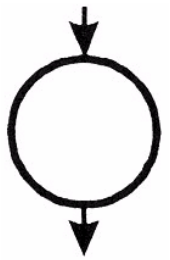
Componente de Caso de Uso	**Descripción**
Número de Caso de Uso o Identificador	Un identificador para este caso de uso.
Nombre de Casos de Uso	El nombre debe ser la meta indicada como un verbo de la frase.
Meta en contexto	Una declaración más detallada de la meta en caso de necesitarla.
Alcance	Corporación / Sistema / Subsistema.

Componente de Caso de Uso	Descripción	
Nivel	Resumen / Tarea primaria / Sub-función.	
Actor primario	Nombre del papel o descripción del agente primario.	
Precondiciones	El estado requerido del sistema antes de que el caso de uso se accione.	
Condiciones Finales exitosas	El estado final del sistema en caso de una terminación acertada de este caso de uso.	
Condiciones Finales Interrumpidas	El estado del sistema si el caso de uso no puede ejecutarse hasta su terminación	
Disparador (Trigger)	La acción que inicia la ejecución del caso de uso.	
Escenario Principal de Éxito	**Paso**	**Acción**
	1	Descripción de la acción, puede ser del actor o del sistema.
	2	Descripción de la acción...
	…	Descripción de la acción...
Extensiones	Condiciones bajo las cuales el escenario principal de éxito variará con una descripción de esas variaciones.	
Sub-variantes	Variaciones que no afectan el flujo principal pero que deben ser consideradas.	
Prioridad	Criticidad.	
Tiempo de Reacción	Tiempo disponible para ejecutar este caso del uso.	
Frecuencia	Cuántas veces se ejecuta este caso de uso.	
Vías para el actor primario	Interactivo / Archivo / Base de Datos	
Actores Secundarios	Otros actores necesitan llevan a cabo este caso de uso.	
Vías a los agentes secundarios	Interactivo / Archivo / Base de Datos	
Fecha de vencimiento	Información del horario.	
Nivel Completo	Caso de Uso identificado (0.1) / Escenario principal definido (0.5) / Todas las extensiones definidas (0.8) / Todos los campos completos (1.0)	
Ediciones pendientes	Ediciones sin resolver que aguardan decisiones.	

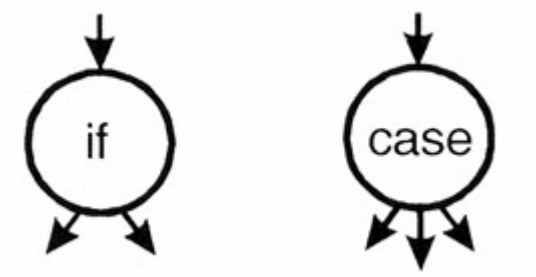
37. Gráfica de control de flujo

Las gráficas documentan las estructuras de control de los módulos. Los módulos de código son convertidos a gráficas, los caminos a través de la gráfica son analizados y los casos de prueba son elaborados como resultado de ese análisis.

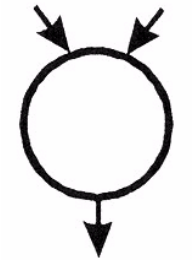
- Bloque de proceso. Es una secuencia de líneas de código de un programa que se ejecuta de manera secuencial de inicio a fin. No se permite ninguna entrada en el bloque excepto al principio. No se permite ninguna salida del bloque excepto en el final. El bloque se inicia una vez, cada declaración dentro de él será ejecutada en secuencia. Los bloques del proceso son representados en los gráficos del control de flujo por una burbuja con una entrada y una salida.



- Punto de decisión. Un punto de decisión es un punto en el módulo en el cual el control de flujo puede cambiar. La mayoría de los puntos de decisión son binarios y son puestos en ejecución por declaraciones if-then-else. De muchas formas, los puntos de decisión son puestos en ejecución por declaraciones del caso. Son representados por una burbuja con una entrada y salidas múltiples.

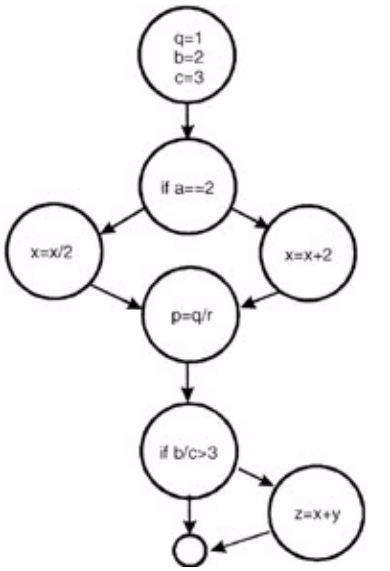


- Punto de unión. Un punto de unión es un punto en el cual los flujos de control se unen.



Ejemplo de código representado por su gráfico de flujo asociado:

```
q=1;
b=2;
c=3;
if (a==2) {x=x+2;}
else {x=x/2;}
p=q/r;
if (b/c>3) {z=x+y;}
```

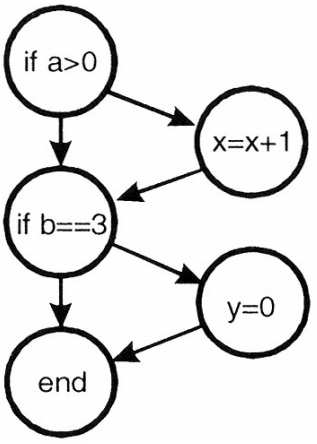


38. Nivel de cobertura

Se utiliza en el flujo de control para definir lo niveles de cobertura. Por cobertura se entiende como el porcentaje de código que se ha probado contra el que está por probarse.

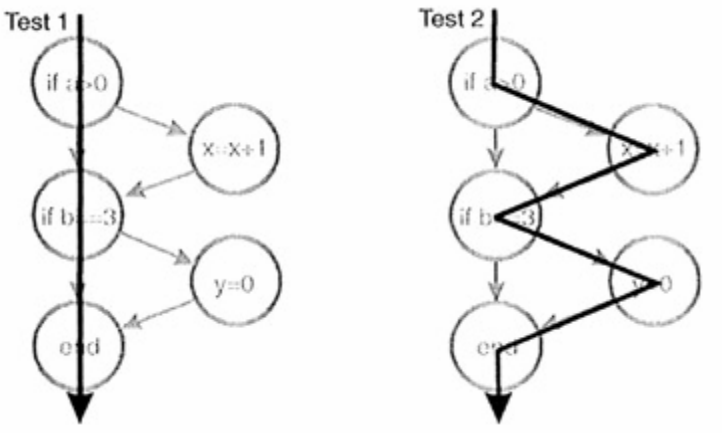
- Nivel 1. El nivel más bajo con “el 100% de las líneas cubiertas”. Significa que cada línea dentro del módulo es ejecutado por una prueba por lo menos una vez. Aunque una sola prueba puede ejecutar todas las líneas del módulo no garantiza que puedan omitirse otros caminos. Es importante recordar otros factores que influyen como la memoria, el espacio en disco, las conexiones, etc. Ejemplo.

```
if (a>0) {x=x+1;}
if (b==3) {y=0;}
```



- Nivel 0. Es el nivel por debajo del “100% de las líneas cubiertas”. El nivel es definido como “pruebe lo que se pruebe se debe dejar probar el resto al usuario”. Para Boris Beizer este nivel es considerado como irresponsable.

- Nivel 2. Este nivel es el “100% de cobertura de decisión”. En este nivel se realizan casos de prueba a cada decisión que tenga un resultado VERDADERO y FALSO por lo menos una vez. Por ejemplo, con el caso (a=2, b=2 y a=4, b=3) bastan con dos casos de prueba para cubrir el 100% de cobertura de decisión.



- Nivel 3. No todas las declaraciones condicionales son tan simples como las que están demostradas previamente. Considere estas declaraciones más complicadas:

```
if (a>0 && c==1) {x=x+1;}
  if (b==3 || d<0) {y=0;}
```

Para ser TRUE, la primera declaración requiere que **a** sea mayor de 0 y **c** igual a 1. El segundo requiere que **b** sea igual a 3 o **d** menos que 0.

En la primera declaración si el valor de **a** fuera fijado en 0 para los propósitos de prueba entonces la parte **c**==1 de la condición no sería probada.

El nivel siguiente de la cobertura del control del flujo es "100% de la cobertura de condición." A este nivel, muchos casos de prueba son escritos para evaluar cada condición que tenga un resultado VERDADERO y FALSO que resuelva una decisión por lo menos una vez.

Este nivel de la cobertura se puede alcanzar con dos casos de la prueba ($a>0$, $c=1$, $b=3$, $d<0$ y $a\leq 0$, $c\neq 1$, $b\neq 3$, $d\geq 0$).

La cobertura de la condición es generalmente mejor que la decisión de cobertura porque cada condición individual se prueba por lo menos una vez mientras que la cobertura de la decisión puede ser alcanzada sin probar cada condición.

- Nivel 4. Considere esta condición:

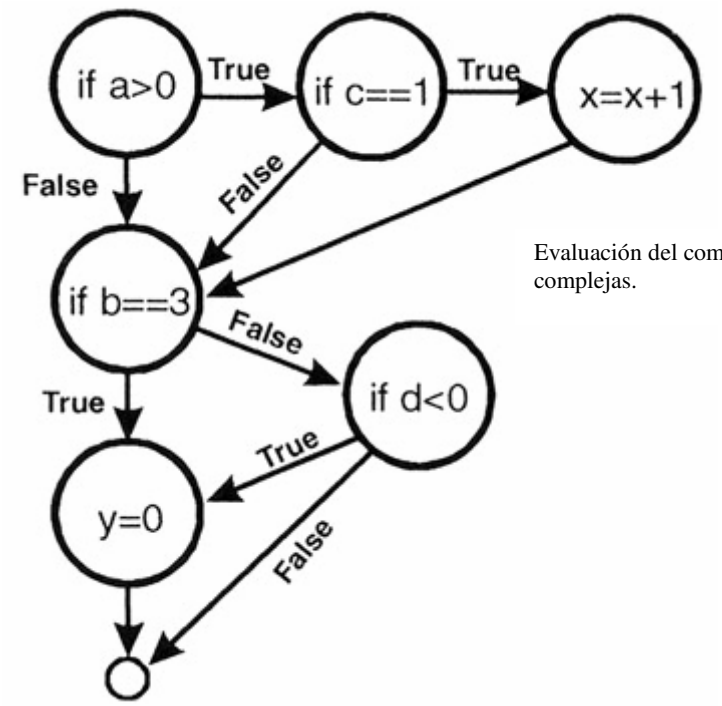
```
if(x&&y) {conditionedStatement;}
          // && Indica Y
```

Podemos alcanzar la condición de cobertura con dos casos de prueba (x=TRUE, y=FALSE y x=FALSE, y=TRUE) pero con esos valores el conditionedStatement nunca será ejecutado. Dada la posible combinación de esas condiciones, será más completo la opción "100% decisión/condición" de cobertura. A este nivel los casos de la prueba se crean para cada condición y cada decisión.

- Nivel 5. Para ser aún más cauteloso, considere cómo el compilador del lenguaje de programación evalúa realmente las condiciones múltiples en una decisión. Utilice ese conocimiento para crear los casos de la prueba que rinden "100% las coberturas múltiples de la condición."

```
if (a>0 && c==1) {x=x+1;}
  if (b==3 || d<0) {y=0;}
```

Será evaluado como:



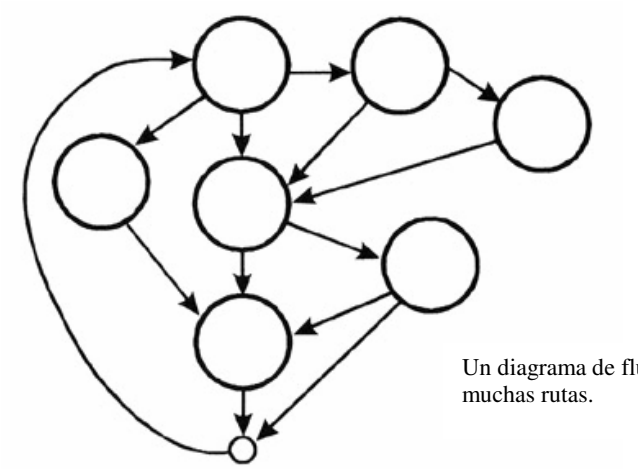
Evaluación del compilador de condiciones complejas.

Este nivel de cobertura se puede alcanzar con cuatro casos de prueba:

a>0, c=1, b=3, d<0
a≤0, c=1, b=3, d≥0
a>0, c≠1, b≠3, d<0
a≤0, c≠1, b≠3, d≥0

Alcanzando 100% de la condición de coberturas múltiples también alcanza cobertura de la decisión, estado de cobertura, y cobertura de decisión/condición. Observe que la cobertura múltiple de la condición no garantiza cobertura de la trayectoria.

- Nivel 7. Finalmente alcanzamos el nivel más alto, que es "100% de cobertura de trayectoria." Para los módulos del código sin iteraciones, el número de las trayectorias generalmente es bastante pequeño que realmente se puede construir un caso de prueba para cada trayectoria. Para los módulos con iteraciones, el número de trayectorias puede ser enorme y plantear así un problema de prueba insuperable.



Un diagrama de flujo interesante con muchas, muchas rutas.

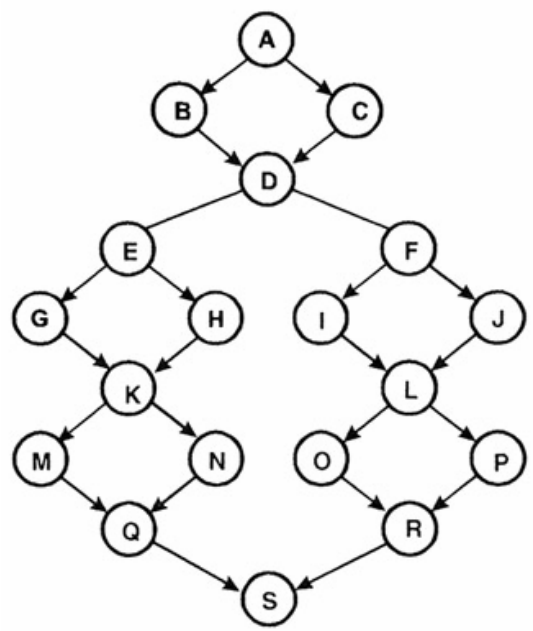
- Nivel 6. Cuando un módulo tiene iteraciones en las trayectorias del código tales que el número de trayectorias es infinito, una importante pero significativa reducción puede ser utilizada limitando la ejecución de la iteración a un número pequeño de casos. El primer caso debe ejecutar cero veces la iteración; lo segundo es ejecutar la iteración una vez, el tercero es ejecutar la iteración **n** veces donde **n** es un número pequeño que representa un valor típico de la iteración; el cuarto es ejecutar la iteración el número máximo de veces **m**. Además que usted puede ser que intente **m-1** y **m+1**.

39. Complejidad ciclomática

McCabe define la complejidad ciclomática de un gráfico como:

$$C = \text{bordes} - \text{nodos} + 2$$

Los bordes son las flechas y los nodos son las burbujas en el gráfico. El siguiente diagrama de control de flujo tiene 24 bordes y 19 nodos para una complejidad de ciclo automático de 24-19+2 = 7.



En algunos casos este cálculo puede ser simplificado. Si todas las decisiones en el diagrama son binarias con lo que hay **p** decisiones binarias, entonces:

$$C = p+1$$

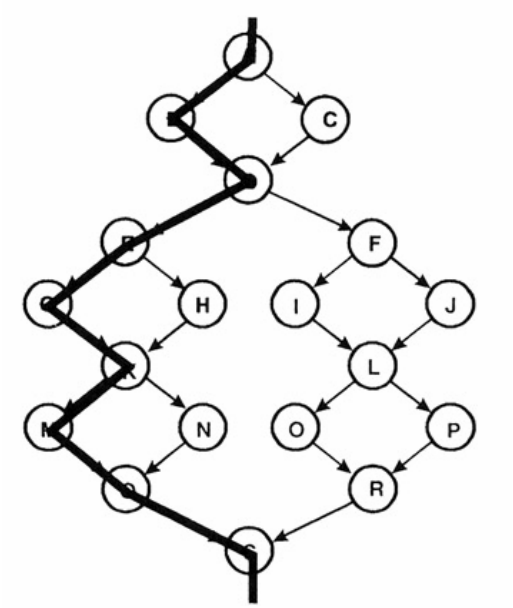
La complejidad ciclimática es exactamente el número de caminos linealmente independientes que contiene un programa y generan todas las rutas posibles a través del módulo. En términos de un diagrama de flujo, cada ruta base atraviesa por lo menos un borde que no hacen las otras rutas.

La técnica de McCabe es empleada para crear C casos de prueba, una para cada camino base. Los caminos base cubren todos los bordes y nodos del diagrama de control de flujo.

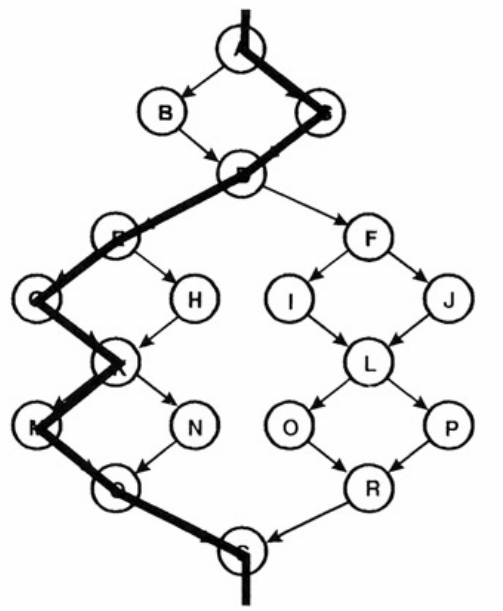
40. Camino base

El proceso para crear un conjunto de caminos base consiste en los siguientes pasos:

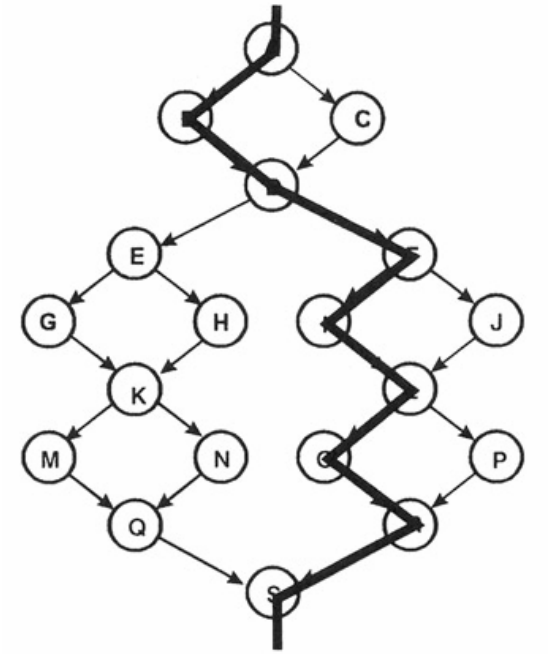
1. Seleccionar un camino como partida. Éste deberá de ser un camino común de ejecución y no se excepción. La selección del camino tendrá la importancia desde una perspectiva de pruebas.



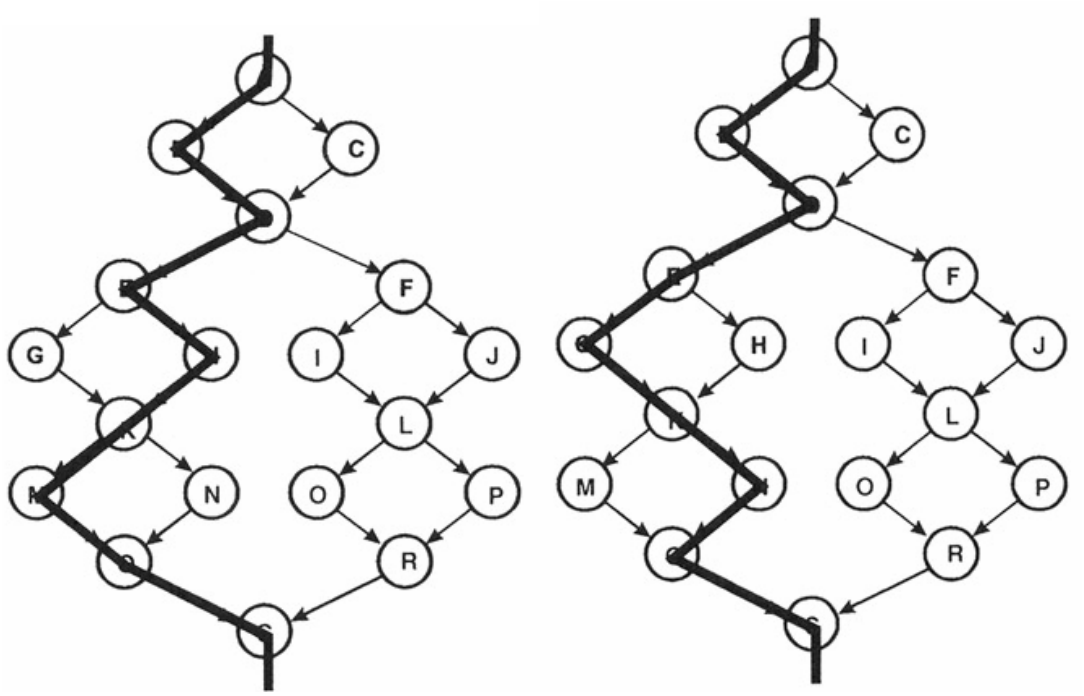
2. Seleccione el siguiente camino, cambiando la salida de la primera decisión del camino con el que se comenzó y tratando de mantener sin cambio el mayor número de las otras decisiones.



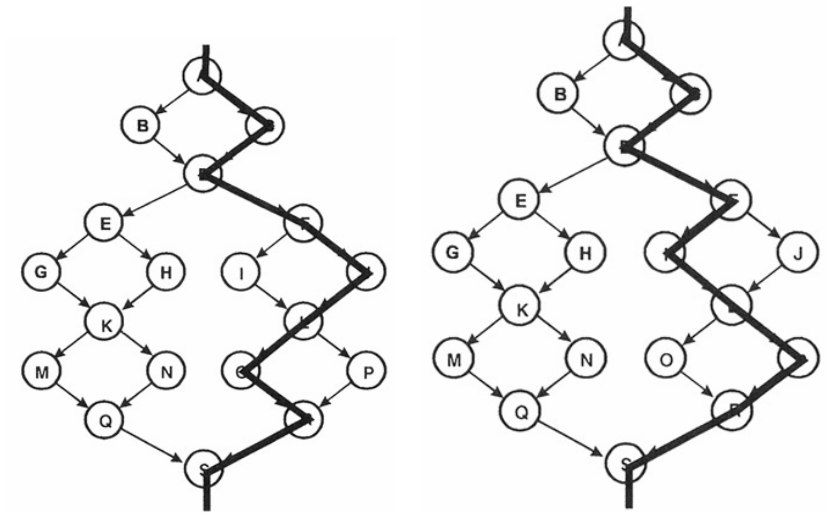
3. para generar el siguiente camino, se comienza nuevamente con el primer camino pero variando la segunda decisión a comparación del primer camino.



4. Para el cuarto camino, se comienza nuevamente como el primer camino pero variando la tercera decisión en vez de la segunda. Continúe variando cada decisión de una por una hasta que el último de la gráfica sea alcanzado.



5. Una vez que todas las decisiones a través del camino base se han abarcado, se procede a la segunda trayectoria y abarcando cada una de las decisiones. Se continúa con este patrón hasta completar todo el camino base.



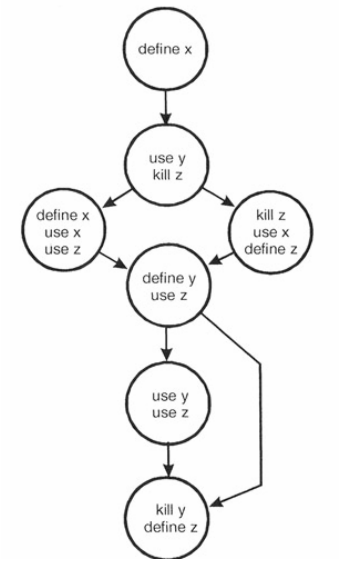
De esta forma se obtienen los siguientes caminos base:

- ABDEGKMQS
- ACDEGKMQS
- ABDFILORS
- ABDEHKMQS
- ABDEGKNQS
- ACDFJLORS

- ACDFILPRS

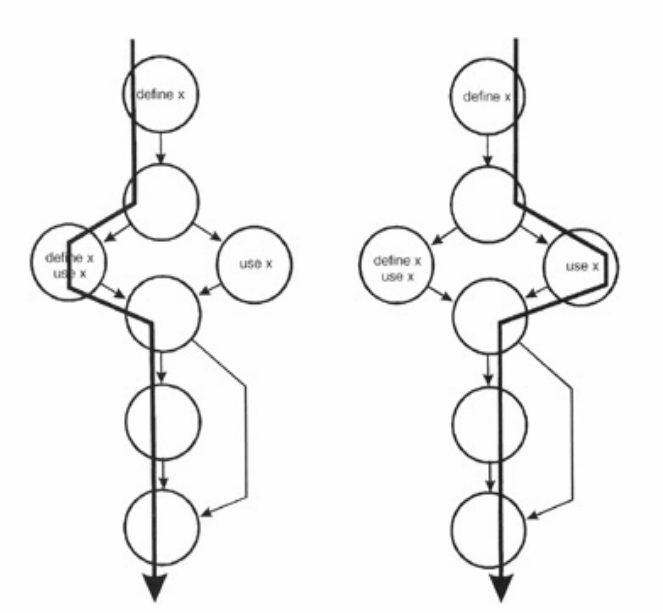
41. Gráfica de flujo de datos

Es similar a la gráfica de control de flujo, pero en este se detalla la definición, uso y destrucción de las variables de cada módulo.



42. Inspección estática de la gráfica de flujo de datos

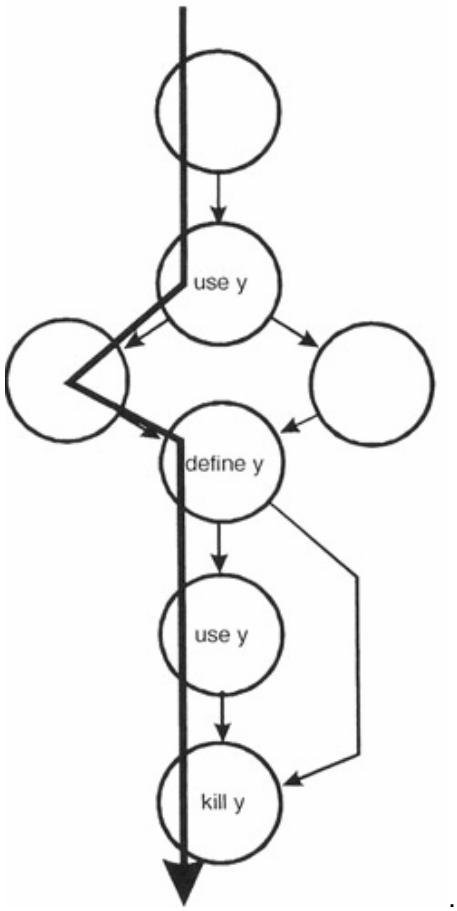
A cada variable dentro del flujo se examinará sus parámetros de definición, uso y destrucción a lo largo de los caminos de control de flujo.



Los parámetros definición, uso y destrucción para **X** (en pares) son:

~ definición	Correcto, un caso normal.
definición-definición	Sospechoso, posiblemente un error de programación.
definición-uso	Correcto, un caso normal.

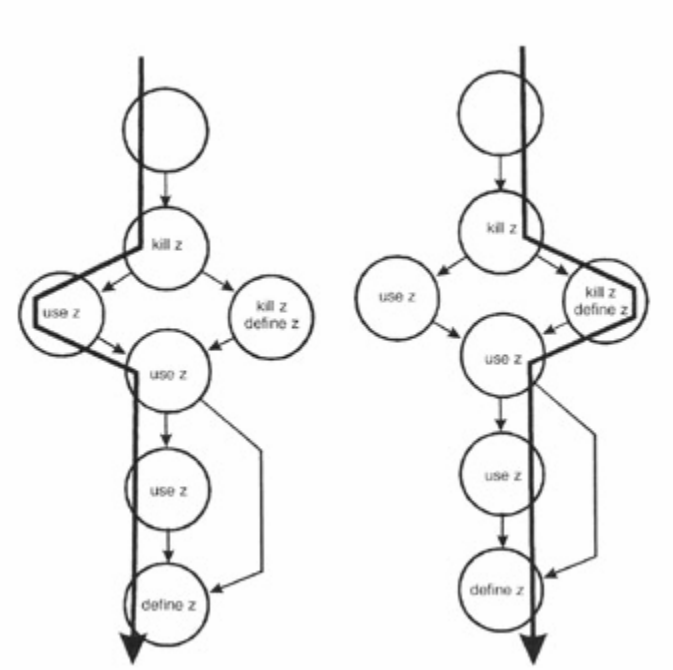
Ahora para la variable **Y**. Note que la primer ramificación del módulo no tiene impacto sobre la variable **Y**



Los parámetros definición, uso y destrucción para **Y** son:

~usar	Es una equivocación importante.
usar-definir	Es aceptable.
definir-usar	Es correcto, un caso normal.
usar-destruir	Aceptable.
definir-destruir	Probablemente es un error de programación.

Ahora para la variable **Z**:



Los parámetros definición, uso y destrucción para **Z** son:

~ destruir	Error de programación.
destruir-usar	Error importante de programación
usar-usar	Correcto, un caso normal.
usar-definir	Aceptable.
destruir-destruir	Probablemente un error de programación.
destruir-definir	Aceptable.
definir-usar	Correcto, un caso normal.

Con la realización del análisis estático se detectaron las siguientes irregularidades:

x: definir-definir
y: ~usar
y: definir-destruir
z: ~destruir
z: destruir-usar
z: destruir-destruir

43. Inspección dinámica del flujo de datos

Detecta mediante la ejecución de casos de prueba algunos de los errores en el flujo de datos que la inspección estática no puede llevar a cabo.

- Seleccionar los suficientes casos de prueba que permitan dar seguimiento a la definición de variables con su uso y del uso con su correspondiente definición.
- Enumerar los módulos de cada camino como lo realiza la prueba de control de flujo.
- Crear un caso de prueba a cada variable para cubrir su definición y uso.

44. Casos de prueba

Los casos de prueba consisten en entradas, salidas y orden de ejecución en condiciones específicas.

Entradas. Se refiere a la entrada de datos que ingresan a un sistema y que pueden provenir de varias fuentes como interfases del sistema, interfaces con dispositivos, archivos, bases de datos, etc.

Salidas. Son los datos que salen del sistema que pueden ser enviados a interfases de sistemas o dispositivos, archivos, bases de datos, pantalla, etc.

Orden de ejecución. Existen dos estilos para diseñar casos de prueba. Los casos de prueba en cascada y los casos de prueba independientes. El primero se encuentra diseñado para que una prueba sirva a la siguiente prueba. Y las últimas son pruebas autosuficientes que no requieren de que se haya realizado una prueba previamente

Según la norma IEEE 829, la especificación de casos de prueba deberá de contar con los siguientes elementos:

1. Identificador del caso de prueba. Un identificador único que le permitirá al documento diferenciarse de los demás documentos.
2. Elementos de prueba. Identifica los elementos y características que serán sometidos a prueba por el caso de prueba.
3. Especificaciones de entrada. Especifica cada entrada requerida por el caso de prueba.
4. Especificaciones de salida. Especifica cada salida esperada después de que se ejecute el caso de prueba.
5. Requerimientos del ambiente. Cualquier elemento especial que se requiera para llevar a cabo el caso de prueba que no haya sido mencionado en las especificaciones del diseño de pruebas.
6. Requerimientos especiales de procedimiento. Se define cualquier *setup*, ejecución o procedimiento especial único para el caso de prueba.
7. Dependencia con otros casos. Lista de casos de prueba que deben de ser ejecutados antes de ejecutar el caso de prueba.

Herramientas

1. Criterios de aceptación. Consiste en proporcionar estándares que deben de ser alcanzados por el sistema para que sea aceptable para el usuario.

2. Análisis de valor límite. Divide al sistema de arriba hacia abajo en segmentos lógicos y con ello limita las pruebas con límites en cada segmento.

3. Gráfica causa efecto. Intenta mostrar los efectos de cada evento procesado con en fin de categorizar eventos por el efecto que ocurrirá como resultado del procesamiento.

4. Checklist. Una serie de preguntas diseñadas para ser utilizadas en la revisión en una determinada área o función.

5. Comparación de código. Identifica la diferencia entre dos versiones del mismo programa.

6. Análisis basado en compilador. Detecta errores de un programa durante el proceso de compilación.

7. Métrica basada en complejidad. Utiliza relaciones para identificar el grado de complejidad de procesamiento del programa.

8. Confirmación / Examinación. Verifica que las condiciones han o no han ocurrido.

9. Análisis de control de flujo. Identifica inconsistencias de procesamiento para identificar problemas lógicos dentro de los programas.

10. *Correctness Proof*. Involucra un conjunto de sentencias o hipótesis que definen si se está realizando un correcto procesamiento. Esas hipótesis son probadas para determinar si las aplicaciones desempeñan el procesamiento de acuerdo con las sentencias.

11. Métricas basadas en cobertura. Utiliza relaciones matemáticas para mostrar qué porcentaje de las aplicaciones han sido cubiertas por el proceso de pruebas. La métrica resultante puede ser utilizada para predecir la efectividad del proceso de pruebas.

12. Diccionario de datos. Genera datos de prueba para verificar la validación de los programas de validación de programas basados en los datos contenidos en el diccionario.

13. Análisis de flujo de datos. Un método para asegurar que los datos usados por el programa han sido definidos adecuadamente y los datos definidos sean utilizados adecuadamente.

14. Pruebas funcionales basadas en diseño. Reconoce qué funciones dentro de una aplicación son necesarias para los requerimientos. Este proceso identifica las funciones basadas en diseño para propósitos de pruebas.

15. Revisiones del diseño. Revisiones conducidas durante el proceso de desarrollo normalmente en concordancia con la metodología de desarrollo. El objetivo primario es asegurar la conformidad con la metodología de desarrollo.

16. Revisiones de escritorio. Proporcionan una evaluación realizada por el programador o analista al elemento de programa lógico antes de ser codificado o diseñado.

17. Prueba de desastre. Son simulaciones de fallas para determinar si el sistema puede ser correctamente recuperado después de la falla.

18. Conjeturas de error. Utiliza la experiencia o el juicio de la gente para determinar a través de conjeturas los errores que muy probablemente puedan ocurrir y a partir de ellos probar para asegurarnos si el sistema puede manejar esas condiciones.

19. Especificaciones ejecutables. Requiere de una interpretación del sistema de alto nivel para escribir las especificaciones. Las especificaciones compiladas tienen menos detalles y precisión a comparación de los programas finales implementados, pero son suficientes para evaluar que las especificaciones se encuentran completas y con el adecuado funcionamiento.

20. Pruebas exhaustivas. Se intenta crear suficientes pruebas para evaluar cada camino y condición de la aplicación.

21. Indagación de hechos. Desempeña los pasos necesarios para obtener hechos de apoyo al proceso de pruebas.

22. Diagrama de flujo. Representación gráfica de la lógica y el flujo de datos de una aplicación.

23. Inspecciones. Revisión paso por paso del producto en el que cada paso es revisado contra una lista con criterios predeterminados.

24. Instrumentación. Mide la función de la estructura de un sistema utilizando contadores u otro instrumento de monitoreo.

25. Facilidad integrada. Permite la introducción de datos de prueba en el ambiente de producción, de forma que las aplicaciones pueden ser probadas al mismo tiempo que se encuentran ejecutándose en producción.

26. Mapeo. Identifica qué partes de un programa son ejecutadas durante una prueba y su frecuencia.

27. Modelado. Método de simulación de las funciones de la aplicación y/o ambiente para determinar si las especificaciones de diseño alcanzan los objetivos del sistema.

28. Operación en paralelo. Se corre la vieja y la nueva versión a la vez para identificar las diferencias entre los dos procesos.

29. Simulación en paralelo. Aproxima los resultados esperados de procesamiento para simular el proceso y determinar si los resultados son razonables.

30. Revisión en pares. Provee una evaluación en pares de la eficiencia, estilo, seguimiento de los estándares. Del producto diseñado para mejorar la calidad del producto.

31. Matriz de riesgos. Se produce una matriz que muestra las relaciones entre los riesgos del sistema, el segmento en el sistema en donde ocurre el riesgo y la ausencia de controles para reducir ese riesgo.

32. Sistema de control, auditoría y revisión de archivos. (SCARF por sus siglas en inglés) Construye un historial de los errores potenciales con la finalidad de comparar problemas en una unidad en un periodo de tiempo y/o compararlo contra otras unidades.

33. Marcador. Identifica áreas en la aplicación que requieren pruebas, a través de un criterio para relacionarlas a problemas.

34. Foto. Muestra el contenido almacenado de una computadora en puntos específicos durante el proceso.

35. Ejecución simbólica. Identifica caminos de procesamiento para probar programas con símbolos y no con datos de prueba.

36. Registros del sistema. Proporciona un rastro y monitorea eventos en un ambiente controlado por un software.

37. Datos de prueba. Crea transacciones para que se utilicen en determinadas funciones de un sistema de computadora.

38. Generados de datos de prueba. Proporciona pruebas de transacciones basadas en parámetros que necesitan probarse.

39. Seguimiento. Sigue y enlista el flujo de procesamiento y datos basados en búsquedas.

40. Programa utilitario. Analiza e imprime el resultado de una prueba a través del uso de un programa de propósito general.

41. Prueba de volumen. Identifica las restricciones del sistema y crea grandes volúmenes de transacciones diseñados para exceder esos límites.

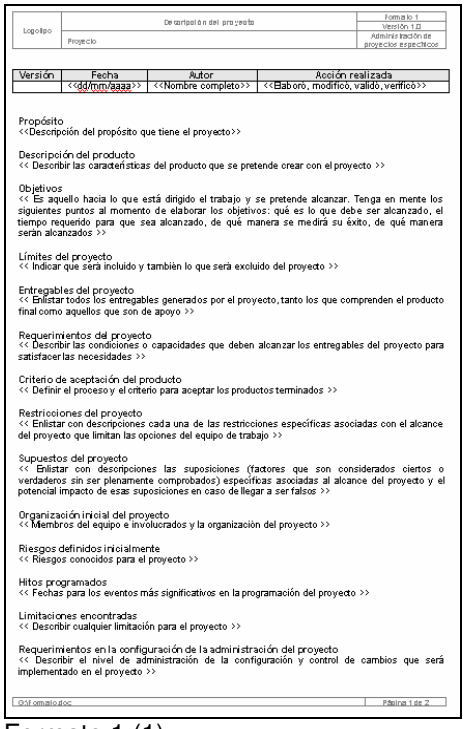
42. Walk-throughs. Lleva al equipo de pruebas a realizar una simulación manual del producto utilizando pruebas de transacción. Se cuestiona al autor del elemento de software y se le solicita que explique su funcionamiento.

Formatos

Esta sección contiene los formatos que son parte de los procesos: Administración de proyectos específicos y Desarrollo y mantenimiento de software.

Clave	Versión	Nombre	Comentarios
Formato 1	1.0	Descripción del proyecto	Describe en detalle los entregables del proyecto y el trabajo requerido para crear esos entregables.
Formato 2	1.0	Plan del proyecto	Documento formal que sirve como guía para la ejecución y control del proyecto.
Formato 3	1.0	Documentación del proceso	Documento que describe la estructura de un proceso.
Formato 4	1.0	Documento de aceptación	Documento que establece la aceptación del cliente de los entregables establecidos en el proyecto.
Formato 5	1.0	Especificación de requerimientos	Se compone de una introducción y una descripción de requerimientos.
Formato 6	1.0	Análisis y diseño	Este documento contiene la descripción textual y gráfica de la estructura de los componentes de software.
Formato 7	1.0	Reporte de actividades	Registro periódico de actividades, fechas de inicio y fin, responsables.
Formato 8	1.0	Reporte de verificación	Registro de participantes, fecha, lugar, duración y defectos encontrados.
Formato 9	1.0	Reporte de validación	Registro de participantes, fecha, lugar, duración y defectos encontrados.
Formato 10	1.0	Plan de pruebas de sistema	Identificación de pruebas requeridas para el cumplimiento de los requerimientos especificados.
Formato 11	1.0	Registro de rastreo	Relación entre los requerimientos, elementos análisis y diseño, componentes y planes de pruebas.
Formato 12	1.0	Plan de pruebas de integración	Descripción que contiene: - el orden de integración de los componentes o subsistemas, guiado por la parte arquitectónica de Análisis y diseño. - pruebas que se aplicarán para verificar la interacción entre los componentes.
Formato 13	1.0	Reporte de pruebas de	Registro de participantes, fecha, lugar,

Clave	Versión	Nombre	Comentarios
		integración	duración y defectos encontrados.
Formato 14	1.0	Reporte de pruebas de sistema	Registro de participantes, fecha, lugar, duración y defectos encontrados.
Formato 15	1.0	Acciones correctivas	Acciones establecidas para corregir una desviación o problema con respecto al cumplimiento del Plan del proyecto y Plan de desarrollo.
Formato 16	1.0	Minuta	Documento que describe el objetivo de las reuniones realizadas, los puntos tratados y los acuerdos.
Formato 17	1.0	Reporte de seguimiento	Contiene el registro del avance de las actividades realizadas incluyendo las llevadas a cabo en el Plan de Manejo de Riesgos. El avance se registra por ciclo, incluyendo fecha de inicio y fin. Contiene el registro periódico de las mediciones como: costo real del proyecto, esfuerzo realizado, cambios implementados y clasificados por tipo, tiempo real invertido, defectos encontrados, tamaño de los productos y trabajo duplicado.
Formato 18	1.0	Solicitud de cambios	Documento en el que se describe el cambio que se desea realizar.



Logotipo	Descripción del proyecto	Formato 1 Versión 1.0
	Proyecto	Administración de proyectos específicos

Versión	Fecha	Autor	Acción realizada
	<<dd/mm/aaaa>>	<<Nombre completo>>	<<Elaboró, modificó, validó, verificó>>

Propósito
<<Descripción del propósito que tiene el proyecto>>

Descripción del producto
<< Describir las características del producto que se pretende crear con el proyecto >>

Objetivos
<< Es aquello hacia lo que está dirigido el trabajo y se pretende alcanzar. Tenga en mente los siguientes puntos al momento de elaborar los objetivos: qué es lo que debe ser alcanzado, el tiempo requerido para que sea alcanzado, de qué manera se medirá su éxito, de qué manera serán alcanzados >>

Límites del proyecto
<< Indicar que será incluido y también lo que será excluido del proyecto >>

Entregables del proyecto
<< Enlistar todos los entregables generados por el proyecto, tanto los que comprenden el producto final como aquellos que son de apoyo >>

Requerimientos del proyecto
<< Describir las condiciones o capacidades que deben alcanzar los entregables del proyecto para satisfacer las necesidades >>

Criterio de aceptación del producto
<< Definir el proceso y el criterio para aceptar los productos terminados >>

Restricciones del proyecto
<< Enlistar con descripciones cada una de las restricciones específicas asociadas con el alcance del proyecto que limitan las opciones del equipo de trabajo >>

Supuestos del proyecto
<< Enlistar con descripciones las suposiciones (factores que son considerados ciertos o verdaderos sin ser plenamente comprobados) específicas asociadas al alcance del proyecto y el potencial impacto de esas suposiciones en caso de llegar a ser falsos >>

Organización inicial del proyecto
<< Miembros del equipo e involucrados y la organización del proyecto >>

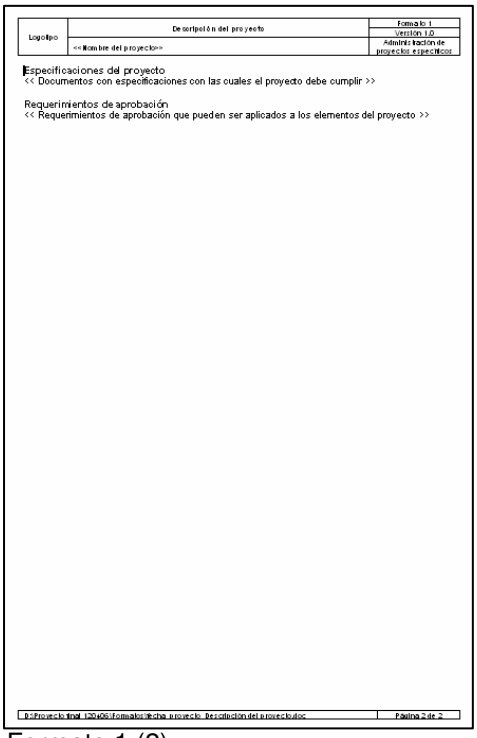
Riesgos definidos inicialmente
<< Riesgos conocidos para el proyecto >>

Hitos programados
<< Fechas para los eventos más significativos en la programación del proyecto >>

Limitaciones encontradas
<< Describir cualquier limitación para el proyecto >>

Requerimientos en la configuración de la administración del proyecto
<< Describir el nivel de administración de la configuración y control de cambios que será implementado en el proyecto >>

Formato 1 (1)

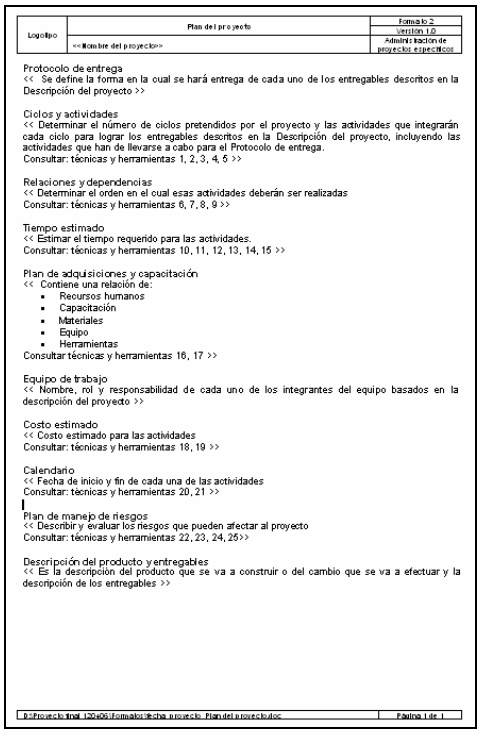


Logotipo	Descripción del proyecto	Formato 1 Versión 1.0
	<<Nombre del proyecto>>	Administración de proyectos específicos

Especificaciones del proyecto
<< Documentos con especificaciones con las cuales el proyecto debe cumplir >>

Requerimientos de aprobación
<< Requerimientos de aprobación que pueden ser aplicados a los elementos del proyecto >>

Formato 1 (2)



Logotipo	Plan del proyecto	Formato 2 Versión 1.0
	<<Nombre del proyecto>>	Administración de proyectos específicos

Protocolo de entrega
<< Se define la forma en la cual se hará entrega de cada uno de los entregables descritos en la Descripción del proyecto >>

Ciclos y actividades
<< Determinar el número de ciclos pretendidos por el proyecto y las actividades que integrarán cada ciclo para lograr los entregables descritos en la Descripción del proyecto, incluyendo las actividades que han de llevarse a cabo para el Protocolo de entrega.
Consultar: técnicas y herramientas 1, 2, 3, 4, 5 >>

Relaciones y dependencias
<< Determinar el orden en el cual esas actividades deberán ser realizadas
Consultar: técnicas y herramientas 6, 7, 8, 9 >>

Tiempo estimado
<< Estimar el tiempo requerido para las actividades.
Consultar: técnicas y herramientas 10, 11, 12, 13, 14, 15 >>

Plan de adquisiciones y capacitación
<< Contiene una relación de:
- Recursos humanos
- Capacitación
- Materiales
- Equipo
- Herramientas

Consultar técnicas y herramientas 16, 17 >>

Equipo de trabajo
<< Nombre, rol y responsabilidad de cada uno de los integrantes del equipo basados en la descripción del proyecto >>

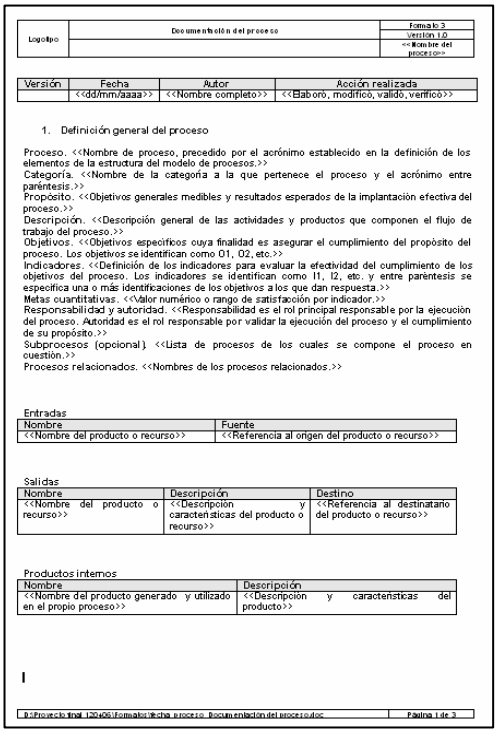
Costo estimado
<< Costo estimado para las actividades
Consultar: técnicas y herramientas 18, 19 >>

Calendario
<< Fecha de inicio y fin de cada una de las actividades
Consultar: técnicas y herramientas 20, 21 >>

Plan de manejo de riesgos
<< Describir y evaluar los riesgos que pueden afectar al proyecto
Consultar: técnicas y herramientas 22, 23, 24, 25>>

Descripción del producto y entregables
<< Es la descripción del producto que se va a construir o del cambio que se va a efectuar y la descripción de los entregables >>

Formato 2 (1)



Logotipo	Documentación del proceso	Formato 3 Versión 1.0
		<<Nombre del proceso>>

Versión	Fecha	Autor	Acción realizada
	<<dd/mm/aaaa>>	<<Nombre completo>>	<<Elaboró, modificó, validó, verificó>>

1. Definición general del proceso

Proceso. <<Nombre de proceso, precedido por el acrónimo establecido en la definición de los elementos de la estructura del modelo de procesos.>>
Categoría. <<Nombre de la categoría a la que pertenece el proceso y el acrónimo entre paréntesis.>>
Propósito. <<Objetivos generales medibles y resultados esperados de la implantación efectiva del proceso.>>
Descripción. <<Descripción general de las actividades y productos que componen el flujo de trabajo del proceso.>>
Objetivos. <<Objetivos específicos cuya finalidad es asegurar el cumplimiento del propósito del proceso. Los objetivos se identifican como O1, O2, etc.>>
Indicadores. <<Definición de los indicadores para evaluar la efectividad del cumplimiento de los objetivos del proceso. Los indicadores se identifican como I1, I2, etc. y entre paréntesis se especifica una o más identificaciones de los objetivos a los que dan respuesta.>>
Metas cuantitativas. <<Valor numérico o rango de satisfacción por indicador.>>
Responsabilidad y autoridad. <<Responsabilidad es el rol principal responsable por la ejecución del proceso. Autoridad es el rol responsable por validar la ejecución del proceso y el cumplimiento de su propósito.>>
Subprocesos (opcional). <<Lista de procesos de los cuales se compone el proceso en cuestión.>>
Procesos relacionados. <<Nombres de los procesos relacionados.>>

Entradas

Nombre	Fuente
<<Nombre del producto o recurso>>	<<Referencia al origen del producto o recurso>>

Salidas

Nombre	Descripción	Destino
<<Nombre del producto o recurso>>	<<Descripción y características del producto o recurso>>	<<Referencia al destinatario del producto o recurso>>

Productos internos

Nombre	Descripción
<<Nombre del producto generado y utilizado en el propio proceso>>	<<Descripción y características del producto>>

Formato 3 (1)

Logotipo	Documentación del proceso	Formato 3 Versión 1.0 <<Nombre del proceso>>

2. Prácticas

Roles involucrados y capacitación. Identifica los roles involucrados y capacitación requerida

Rol	Abreviatura	Capacitación
<<Nombre del rol>>	<<Abreviatura del rol>>	<<Capacitación requerida por el rol para poder ejecutar el proceso>>

Actividades. Se asocian a los objetivos y describen las tareas y roles responsables.

Rol	Descripción
A1 <<Nombre de la actividad (01, 02, ...)>>	
<<Abrev. del (de los) rol (es)>>	A1.1 <<Descripción de la tarea 1. Si la actividad es una verificación o una validación se hará una referencia a la identificación de la misma.>>
	A1.2 <<Descripción de la tarea 2>>
A2 <<Nombre de la actividad (01, 02, ...)>>	
	A2.1 <<Descripción de la tarea 1>>
	A2.2 <<Descripción de la tarea 2>>

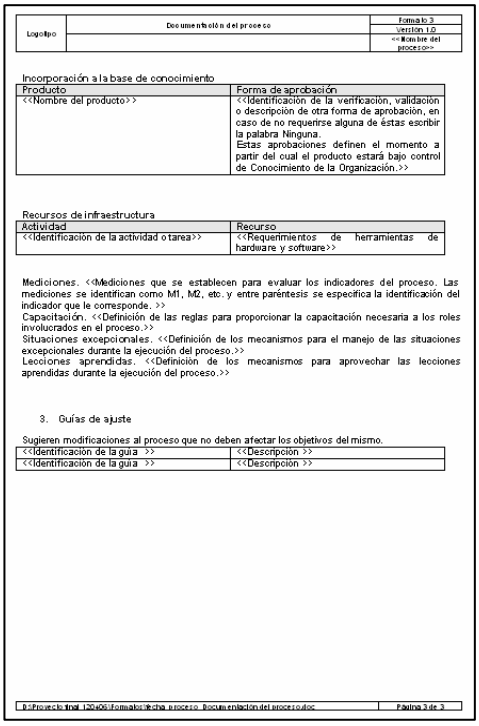
Diagrama de flujo de trabajo. Diagrama de actividades de UML, donde se especifican las actividades del flujo de trabajo y los productos.

Verificaciones y validaciones.
<<Se definen las verificaciones y validaciones asociadas a los productos generados en las actividades que se mencionan.
En la verificación como en la validación se identifican los defectos que deben corregirse antes de continuar con las actividades posteriores. La validación de un producto puede ser interna (dentro de la organización) o externa (por el cliente) con la finalidad de obtener su autorización.
Se recomienda que las validaciones se efectúen una vez que las verificaciones asociadas al producto sean realizadas.>>

Verificación o validación	Actividad	Producto	Rol	Descripción
<<Identificación de la verificación o validación \Ver1 o \Val1>>	<<Identificación de la tarea>>	<<Nombre del producto>>	<<Abreviatura del rol responsable de realizar la verificación o validación>>	<<Descripción de la verificación o validación que se hará del producto>>
<<\Ver2 o \Val2>>				

D:\Proyecto final 120x06\Formatos\fecha proceso Documentación del proceso.doc | Página 2 de 3

Formato 3 (2)



Logotipo	Documentación del proceso	Formato 3 Versión 1.0 <<Nombre del proceso>>

Incorporación a la base de conocimiento

Producto	Forma de aprobación
<<Nombre del producto>>	<<Identificación de la verificación, validación o descripción de otra forma de aprobación, en caso de no requerirse alguna de éstas escribir la palabra Ninguna. Estas aprobaciones definen el momento a partir del cual el producto estará bajo control de Conocimiento de la Organización.>>

Recursos de infraestructura

Actividad	Recurso
<<Identificación de la actividad o tarea>>	<<Requerimientos de herramientas de hardware y software>>

Mediciones. <<Mediciones que se establecen para evaluar los indicadores del proceso. Las mediciones se identifican como M1, M2, etc. y entre paréntesis se especifica la identificación del indicador que le corresponde. >>
Capacitación. <<Definición de las reglas para proporcionar la capacitación necesaria a los roles involucrados en el proceso.>>
Situaciones excepcionales. <<Definición de los mecanismos para el manejo de las situaciones excepcionales durante la ejecución del proceso.>>
Lecciones aprendidas. <<Definición de los mecanismos para aprovechar las lecciones aprendidas durante la ejecución del proceso.>>

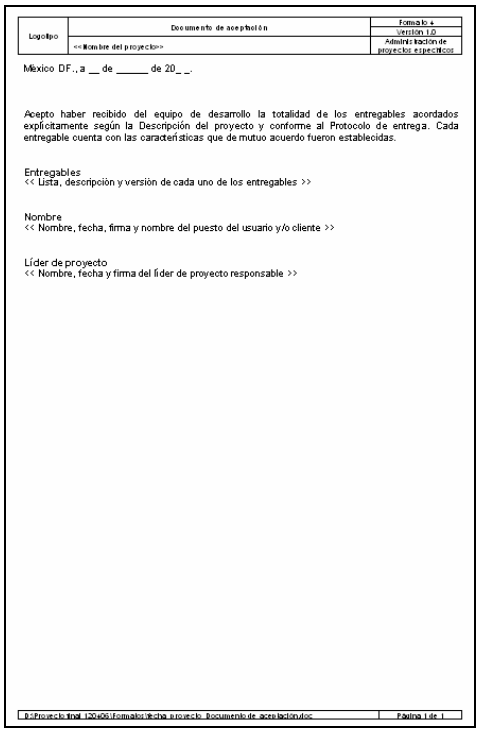
3. Guías de ajuste

Sugieren modificaciones al proceso que no deben afectar los objetivos del mismo.

<<Identificación de la guía >>	<<Descripción >>
<<Identificación de la guía >>	<<Descripción >>

D:\Proyecto final 120x06\Formatos\fecha proceso Documentación del proceso.doc | Página 3 de 3

Formato 3 (3)



Logotipo	Documento de aceptación	Formato 4 Versión 1.0 Administración de proyectos específicos
	<<Nombre del proyecto>>	

México D.F., a __ de _____ de 20_ _.

Acepto haber recibido del equipo de desarrollo la totalidad de los entregables acordados explícitamente según la Descripción del proyecto y conforme al Protocolo de entrega. Cada entregable cuenta con las características que de mutuo acuerdo fueron establecidas.

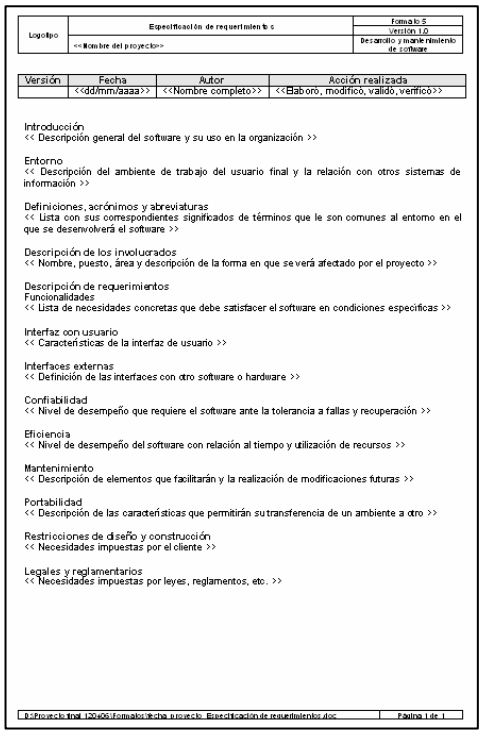
Entregables
<< Lista, descripción y versión de cada uno de los entregables >>

Nombre
<< Nombre, fecha, firma y nombre del puesto del usuario y/o cliente >>

Líder de proyecto
<< Nombre, fecha y firma del líder de proyecto responsable >>

D:\Proyecto final 120x06\Formatos\fecha proyecto Documento de aceptación.doc | Página 1 de 1

Formato 4 (1)



Logotipo	Especificación de requerimientos	Formato 5 Versión 1.0 Desarrollo y mantenimiento de software
	<<Nombre del proyecto>>	

Versión	Fecha	Autor	Acción realizada
	<<dd/mm/aaaa>>	<<Nombre completo>>	<<Elaboró, modificó, validó, verificó>>

Introducción
<< Descripción general del software y su uso en la organización >>

Entorno
<< Descripción del ambiente de trabajo del usuario final y la relación con otros sistemas de información >>

Definiciones, acrónimos y abreviaturas
<< Lista con sus correspondientes significados de términos que le son comunes al entorno en el que se desenvolverá el software >>

Descripción de los involucrados
<< Nombre, puesto, área y descripción de la forma en que se verá afectado por el proyecto >>

Descripción de requerimientos
Funcionalidades
<< Lista de necesidades concretas que debe satisfacer el software en condiciones específicas >>

Interfaz con usuario
<< Características de la interfaz de usuario >>

Interfaces externas
<< Definición de las interfaces con otro software o hardware >>

Confiabilidad
<< Nivel de desempeño que requiere el software ante la tolerancia a fallas y recuperación >>

Eficiencia
<< Nivel de desempeño del software con relación al tiempo y utilización de recursos >>

Mantenimiento
<< Descripción de elementos que facilitarán y la realización de modificaciones futuras >>

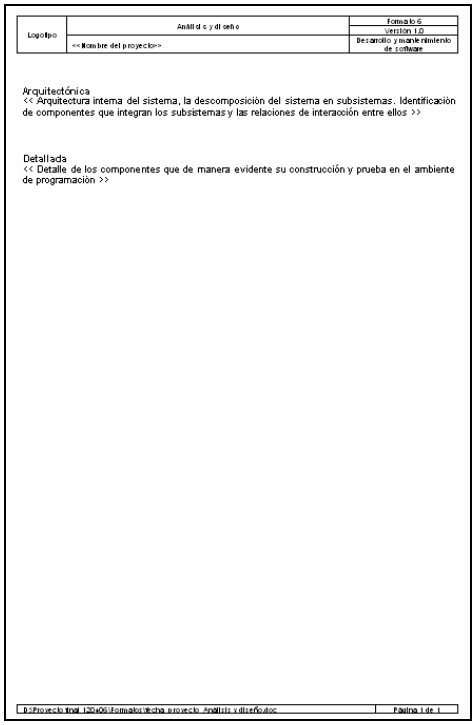
Portabilidad
<< Descripción de las características que permitirán su transferencia de un ambiente a otro >>

Restricciones de diseño y construcción
<< Necesidades impuestas por el cliente >>

Legales y reglamentarios
<< Necesidades impuestas por leyes, reglamentos, etc. >>

D:\Proyecto final 120x06\Formatos\fecha proyecto Especificación de requerimientos.doc | Página 1 de 1

Formato 5 (1)



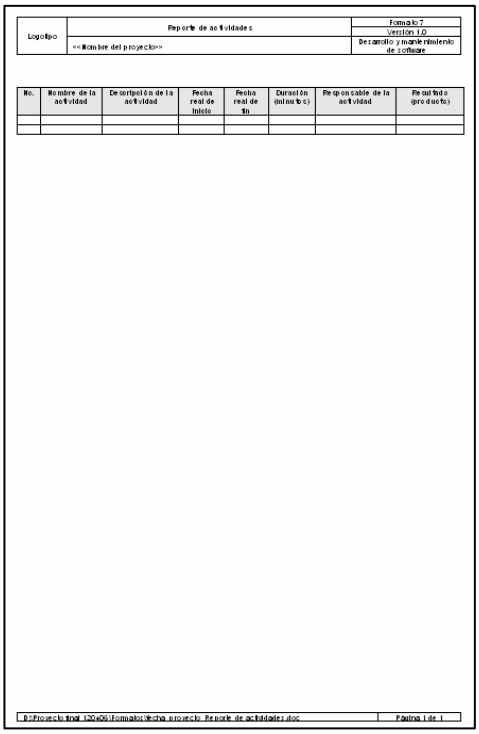
Logotipo	Análisis y diseño	Formato 6
		Versión 1.0
	<< Nombre del proyecto>>	Desarrollo y mantenimiento de software

Arquitectónica
<< Arquitectura interna del sistema, la descomposición del sistema en subsistemas. Identificación de componentes que integran los subsistemas y las relaciones de interacción entre ellos >>

Detallada
<< Detalle de los componentes que de manera evidente su construcción y prueba en el ambiente de programación >>

D:\Proyecto final 120x06\Formatos\fecha proyecto Analisis y diseño.doc | Página 1 de 1

Formato 6 (1)

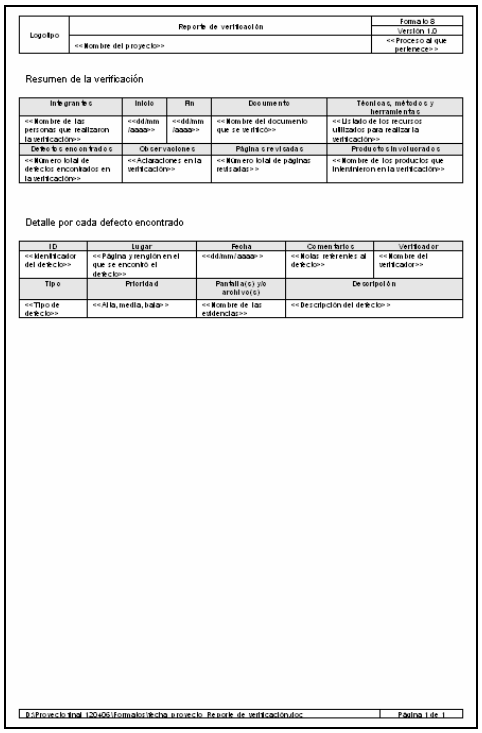


Logotipo	Reporte de actividades	Formato 7
		Versión 1.0
	<< Nombre del proyecto>>	Desarrollo y mantenimiento de software

No.	Nombre de la actividad	Descripción de la actividad	Fecha real de inicio	Fecha real de fin	Duración (minutos)	Responsable de la actividad	Resultado (producto)

D:\Proyecto final 120x06\Formatos\fecha proyecto Reporte de actividades.doc | Página 1 de 1

Formato 7 (1)



Logotipo	Reporte de verificación	Formato 8
		Versión 1.0
	<< Nombre del proyecto>>	<< Proceso al que pertenece>>

Resumen de la verificación

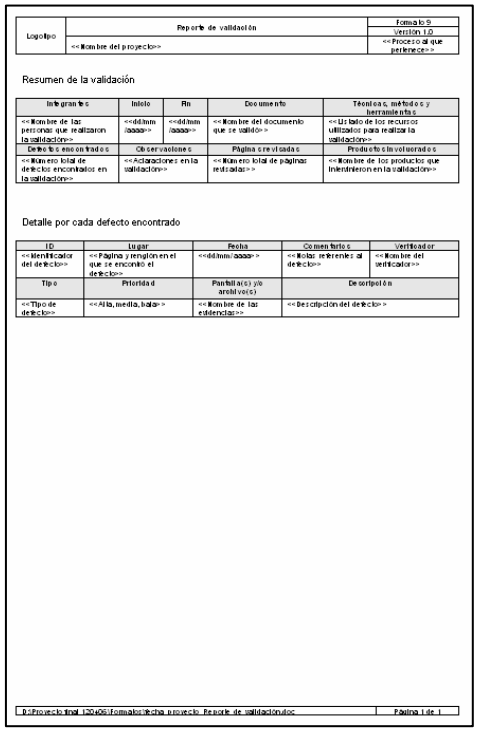
Integrantes	Inicio	Fin	Documento	Técnicas, métodos y herramientas
<<Nombre de las personas que realizaron la verificación>>	<<dd/mm /aaaa>>	<<dd/mm /aaaa>>	<<Nombre del documento que se verificó>>	<<Listado de los recursos utilizados para realizar la verificación>>
Defectos encontrados	Observaciones		Página s revisadas	Productos involucrados
<<Número total de defectos encontrados en la verificación>>	<<Aclaraciones en la verificación>>		<<Número total de páginas revisadas>>	<<Nombre de los productos que intervinieron en la verificación>>

Detalle por cada defecto encontrado

ID	Lugar	Fecha	Comentarios	Verificador
<<Identificador del defecto>>	<<Página y renglón en el que se encontró el defecto>>	<<dd/mm/aaaa>>	<<Notas referentes al defecto>>	<<Nombre del verificador>>
Tipo	Prioridad	Pantalla(s) y/o archivo(s)	Descripción	
<<Tipo de defecto>>	<<Alta, media, baja>>	<<Nombre de las evidencias>>	<<Descripción del defecto>>	

D:\Proyecto final 120x06\Formatos\fecha proyecto Reporte de verificacion.doc | Página 1 de 1

Formato 8 (1)



Logotipo	Reporte de validación	Formato 9
		Versión 1.0
	<< Nombre del proyecto>>	<< Proceso al que pertenece>>

Resumen de la validación

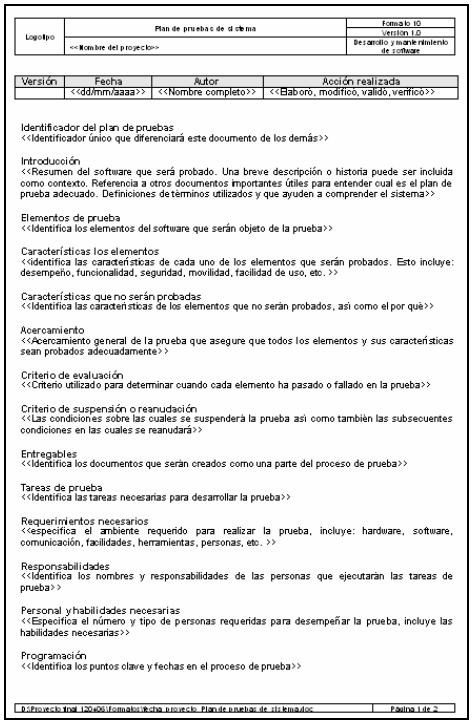
Integrantes	Inicio	Fin	Documento	Técnicas, métodos y herramientas
<<Nombre de las personas que realizaron la validación>>	<<dd/mm /aaaa>>	<<dd/mm /aaaa>>	<<Nombre del documento que se validó>>	<<Listado de los recursos utilizados para realizar la validación>>
Defectos encontrados	Observaciones		Página s revisadas	Productos involucrados
<<Número total de defectos encontrados en la validación>>	<<Aclaraciones en la validación>>		<<Número total de páginas revisadas>>	<<Nombre de los productos que intervinieron en la validación>>

Detalle por cada defecto encontrado

ID	Lugar	Fecha	Comentarios	Verificador
<<Identificador del defecto>>	<<Página y renglón en el que se encontró el defecto>>	<<dd/mm/aaaa>>	<<Notas referentes al defecto>>	<<Nombre del verificador>>
Tipo	Prioridad	Pantalla(s) y/o archivo(s)	Descripción	
<<Tipo de defecto>>	<<Alta, media, baja>>	<<Nombre de las evidencias>>	<<Descripción del defecto>>	

D:\Proyecto final 120x06\Formatos\fecha proyecto Reporte de validacion.doc | Página 1 de 1

Formato 9 (1)



Logotipo	Plan de pruebas de sistema	Formato 10 Versión 1.0
	<<Nombre del proyecto>>	Desarrollo y mantenimiento de software

Versión	Fecha	Autor	Acción realizada
	<<dd/mm/aaaa>>	<<Nombre completo>>	<<Elaboró, modificó, validó, verificó>>

Identificador del plan de pruebas
<<Identificador único que diferenciará este documento de los demás>>

Introducción
<<Resumen del software que será probado. Una breve descripción o historia puede ser incluida como contexto. Referencia a otros documentos importantes útiles para entender cual es el plan de prueba adecuado. Definiciones de términos utilizados y que ayuden a comprender el sistema>>

Elementos de prueba
<<Identifica los elementos del software que serán objeto de la prueba>>

Características los elementos
<<Identifica las características de cada uno de los elementos que serán probados. Esto incluye: desempeño, funcionalidad, seguridad, movilidad, facilidad de uso, etc. >>

Características que no serán probadas
<<Identifica las características de los elementos que no serán probados, así como el por què>>

Acercamiento
<<Acercamiento general de la prueba que asegure que todos los elementos y sus características sean probados adecuadamente>>

Criterio de evaluación
<<Criterio utilizado para determinar cuando cada elemento ha pasado o fallado en la prueba>>

Criterio de suspensión o reanudación
<<Las condiciones sobre las cuales se suspenderá la prueba así como también las subsecuentes condiciones en las cuales se reanudará>>

Entregables
<<Identifica los documentos que serán creados como una parte del proceso de prueba>>

Tareas de prueba
<<Identifica las tareas necesarias para desarrollar la prueba>>

Requerimientos necesarios
<<especifica el ambiente requerido para realizar la prueba, incluye: hardware, software, comunicación, facilidades, herramientas, personas, etc. >>

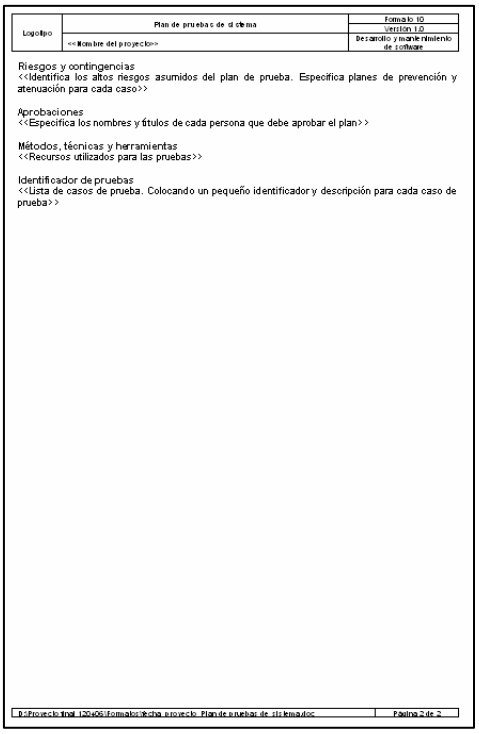
Responsabilidades
<<Identifica los nombres y responsabilidades de las personas que ejecutarán las tareas de prueba>>

Personal y habilidades necesarias
<<Especifica el número y tipo de personas requeridas para desempeñar la prueba, incluye las habilidades necesarias>>

Programación
<<Identifica los puntos clave y fechas en el proceso de prueba>>

D:\Proyecto final 120x06\Formatos\fecha proyecto. Plan de pruebas de sistema.doc | Página 1 de 2

Formato 10 (1)



Logotipo	Plan de pruebas de sistema	Formato 10 Versión 1.0
	<<Nombre del proyecto>>	Desarrollo y mantenimiento de software

Riesgos y contingencias
<<Identifica los altos riesgos asumidos del plan de prueba. Especifica planes de prevención y atenuación para cada caso>>

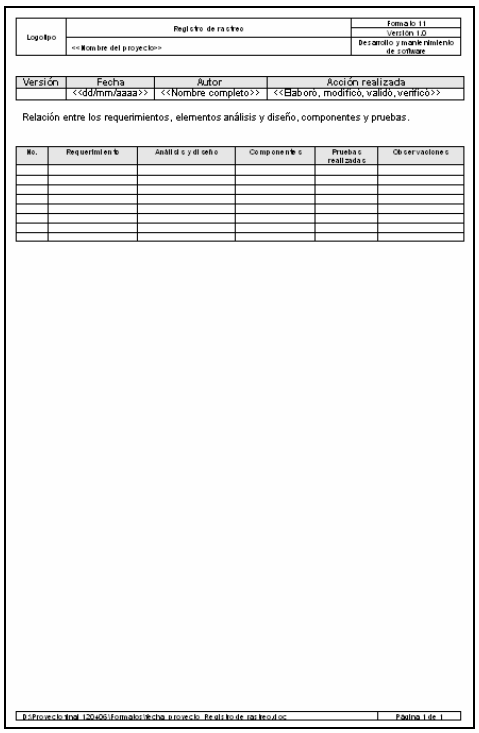
Aprobaciones
<<Especifica los nombres y títulos de cada persona que debe aprobar el plan>>

Métodos, técnicas y herramientas
<<Recursos utilizados para las pruebas>>

Identificador de pruebas
<<Lista de casos de prueba. Colocando un pequeño identificador y descripción para cada caso de prueba>>

D:\Proyecto final 120x06\Formatos\fecha proyecto. Plan de pruebas de sistema.doc | Página 2 de 2

Formato 10 (2)



Logotipo	Registro de rastreo	Formato 11 Versión 1.0
	<<Nombre del proyecto>>	Desarrollo y mantenimiento de software

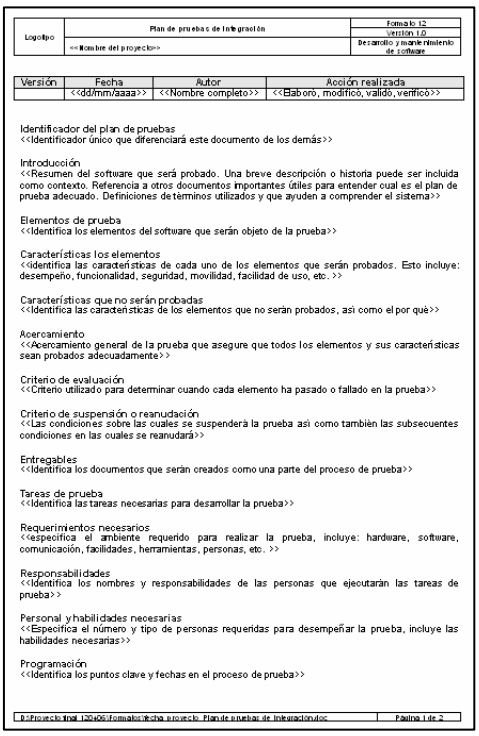
Versión	Fecha	Autor	Acción realizada
	<<dd/mm/aaaa>>	<<Nombre completo>>	<<Elaboró, modificó, validó, verificó>>

Relación entre los requerimientos, elementos análisis y diseño, componentes y pruebas.

No.	Requerimiento	Análisis y diseño	Componentes	Pruebas realizadas	Observaciones

D:\Proyecto final 120x06\Formatos\fecha proyecto. Registro de rastreo.doc | Página 1 de 1

Formato 11 (1)



Logotipo	Plan de pruebas de Integración	Formato 12 Versión 1.0
	<<Nombre del proyecto>>	Desarrollo y mantenimiento de software

Versión	Fecha	Autor	Acción realizada
	<<dd/mm/aaaa>>	<<Nombre completo>>	<<Elaboró, modificó, validó, verificó>>

Identificador del plan de pruebas
<<Identificador único que diferenciará este documento de los demás>>

Introducción
<<Resumen del software que será probado. Una breve descripción o historia puede ser incluida como contexto. Referencia a otros documentos importantes útiles para entender cual es el plan de prueba adecuado. Definiciones de términos utilizados y que ayuden a comprender el sistema>>

Elementos de prueba
<<Identifica los elementos del software que serán objeto de la prueba>>

Características los elementos
<<Identifica las características de cada uno de los elementos que serán probados. Esto incluye: desempeño, funcionalidad, seguridad, movilidad, facilidad de uso, etc. >>

Características que no serán probadas
<<Identifica las características de los elementos que no serán probados, así como el por què>>

Acercamiento
<<Acercamiento general de la prueba que asegure que todos los elementos y sus características sean probados adecuadamente>>

Criterio de evaluación
<<Criterio utilizado para determinar cuando cada elemento ha pasado o fallado en la prueba>>

Criterio de suspensión o reanudación
<<Las condiciones sobre las cuales se suspenderá la prueba así como también las subsecuentes condiciones en las cuales se reanudará>>

Entregables
<<Identifica los documentos que serán creados como una parte del proceso de prueba>>

Tareas de prueba
<<Identifica las tareas necesarias para desarrollar la prueba>>

Requerimientos necesarios
<<especifica el ambiente requerido para realizar la prueba, incluye: hardware, software, comunicación, facilidades, herramientas, personas, etc. >>

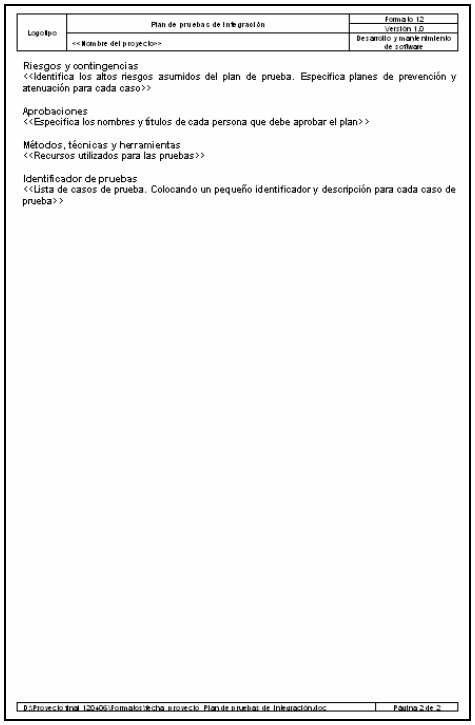
Responsabilidades
<<Identifica los nombres y responsabilidades de las personas que ejecutarán las tareas de prueba>>

Personal y habilidades necesarias
<<Especifica el número y tipo de personas requeridas para desempeñar la prueba, incluye las habilidades necesarias>>

Programación
<<Identifica los puntos clave y fechas en el proceso de prueba>>

D:\Proyecto final 120x06\Formatos\fecha proyecto. Plan de pruebas de Integración.doc | Página 1 de 2

Formato 12 (1)



Logotipo	Plan de pruebas de integración	Formato 12 Versión 1.0
	<<Nombre del proyecto>>	Desarrollo y mantenimiento de software

Riesgos y contingencias
<<Identifica los altos riesgos asumidos del plan de prueba. Especifica planes de prevención y atenuación para cada caso>>

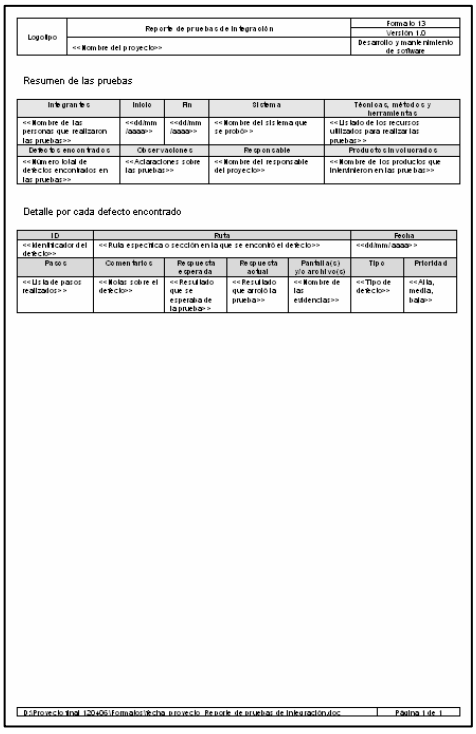
Aprobaciones
<<Especifica los nombres y títulos de cada persona que debe aprobar el plan>>

Métodos, técnicas y herramientas
<<Recursos utilizados para las pruebas>>

Identificador de pruebas
<<Lista de casos de prueba. Colocando un pequeño identificador y descripción para cada caso de prueba>>

D:\Proyecto final 120406\Formatos\fecha proyecto. Plan de pruebas de Integración.doc	Página 2 de 2

Formato 12 (2)



Logotipo	Reporte de pruebas de Integración	Formato 13 Versión 1.0
	<<Nombre del proyecto>>	Desarrollo y mantenimiento de software

Resumen de las pruebas

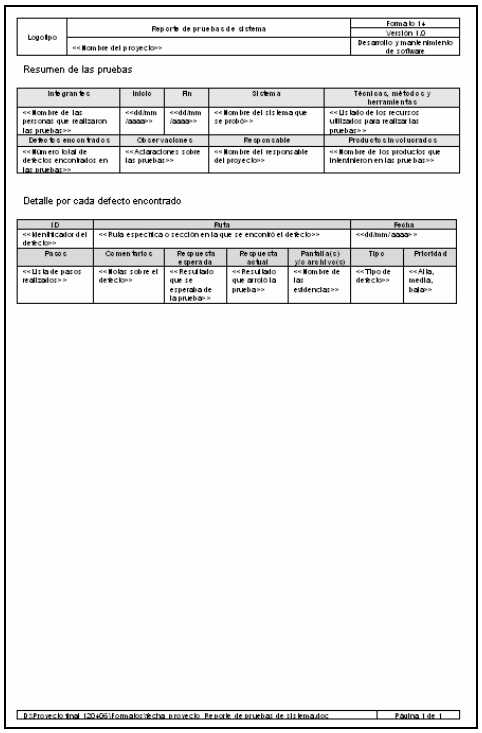
Integrantes	Inicio	Fin	Sistema	Técnicas, métodos y herramientas
<<Nombre de las personas que realizaron las pruebas>>	<<dd/mm /aaaa>>	<<dd/mm /aaaa>>	<<Nombre del sistema que se probó>>	<<Listado de los recursos utilizados para realizar las pruebas>>
Defectos encontrados	**Observaciones**		**Responsable**	**Productos involucrados**
<<Número total de defectos encontrados en las pruebas>>	<<Aclaraciones sobre las pruebas>>		<<Nombre del responsable del proyecto>>	<<Nombre de los productos que intervinieron en las pruebas>>

Detalle por cada defecto encontrado

ID	Ruta				Fecha	
<<Identificador del defecto>>	<<Ruta específica o sección en la que se encontró el defecto>>				<<dd/mm/aaaa>>	
Pasos	**Comentarios**	**Respuesta esperada**	**Respuesta actual**	**Pantalla(s) y/o archivo(s)**	**Tipo**	**Prioridad**
<<Lista de pasos realizados>>	<<Notas sobre el defecto>>	<<Resultado que se esperaba de la prueba>>	<<Resultado que arrojó la prueba>>	<<Nombre de las evidencias>>	<<Tipo de defecto>>	<<Alta, media, baja>>

D:\Proyecto final 120406\Formatos\fecha proyecto. Reporte de pruebas de Integración.doc	Página 1 de 1

Formato 13 (1)



Logotipo	Reporte de pruebas de sistema	Formato 14 Versión 1.0
	<<Nombre del proyecto>>	Desarrollo y mantenimiento de software

Resumen de las pruebas

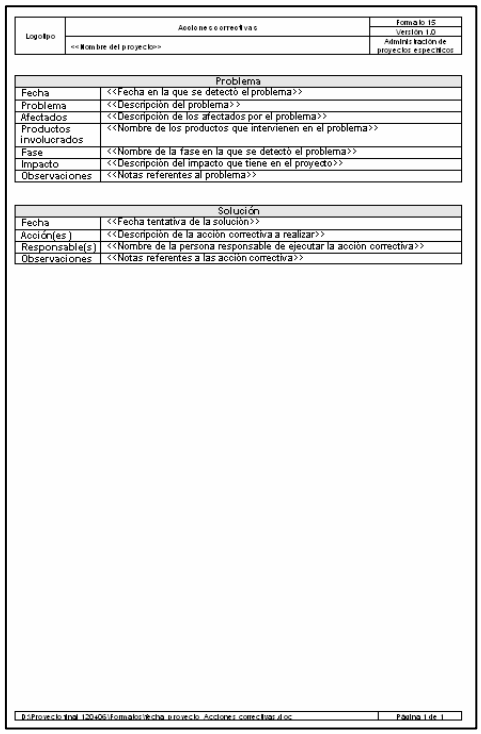
Integrantes	Inicio	Fin	Sistema	Técnicas, métodos y herramientas
<<Nombre de las personas que realizaron las pruebas>>	<<dd/mm /aaaa>>	<<dd/mm /aaaa>>	<<Nombre del sistema que se probó>>	<<Listado de los recursos utilizados para realizar las pruebas>>
Defectos encontrados	**Observaciones**		**Responsable**	**Productos involucrados**
<<Número total de defectos encontrados en las pruebas>>	<<Aclaraciones sobre las pruebas>>		<<Nombre del responsable del proyecto>>	<<Nombre de los productos que intervinieron en las pruebas>>

Detalle por cada defecto encontrado

ID	Ruta				Fecha	
<<Identificador del defecto>>	<<Ruta específica o sección en la que se encontró el defecto>>				<<dd/mm/aaaa>>	
Pasos	**Comentarios**	**Respuesta esperada**	**Respuesta actual**	**Pantalla(s) y/o archivo(s)**	**Tipo**	**Prioridad**
<<Lista de pasos realizados>>	<<Notas sobre el defecto>>	<<Resultado que se esperaba de la prueba>>	<<Resultado que arrojó la prueba>>	<<Nombre de las evidencias>>	<<Tipo de defecto>>	<<Alta, media, baja>>

D:\Proyecto final 120406\Formatos\fecha proyecto. Reporte de pruebas de sistema.doc	Página 1 de 1

Formato 14 (1)



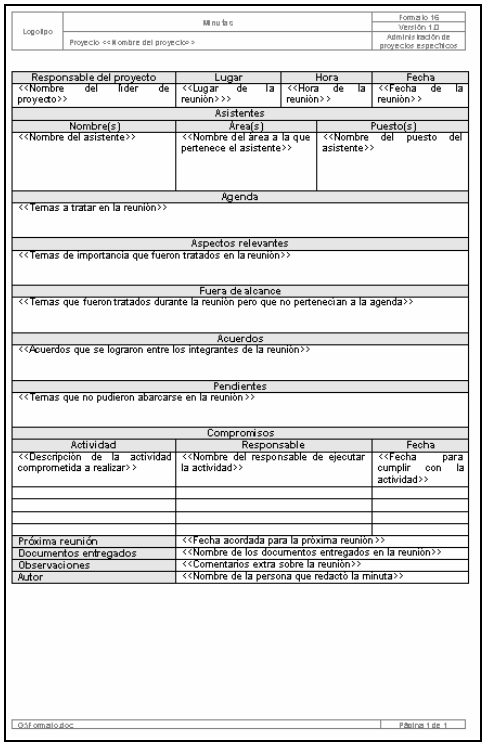
Logotipo	Acciones correctivas	Formato 15 Versión 1.0
	<<Nombre del proyecto>>	Administración de proyectos específicos

Problema	
Fecha	<<Fecha en la que se detectó el problema>>
Problema	<<Descripción del problema>>
Afectados	<<Descripción de los afectados por el problema>>
Productos involucrados	<<Nombre de los productos que intervienen en el problema>>
Fase	<<Nombre de la fase en la que se detectó el problema>>
Impacto	<<Descripción del impacto que tiene en el proyecto>>
Observaciones	<<Notas referentes al problema>>

Solución	
Fecha	<<Fecha tentativa de la solución>>
Acción(es)	<<Descripción de la acción correctiva a realizar>>
Responsable(s)	<<Nombre de la persona responsable de ejecutar la acción correctiva>>
Observaciones	<<Notas referentes a las acción correctiva>>

D:\Proyecto final 120406\Formatos\fecha proyecto. Acciones correctivas.doc	Página 1 de 1

Formato 15 (1)

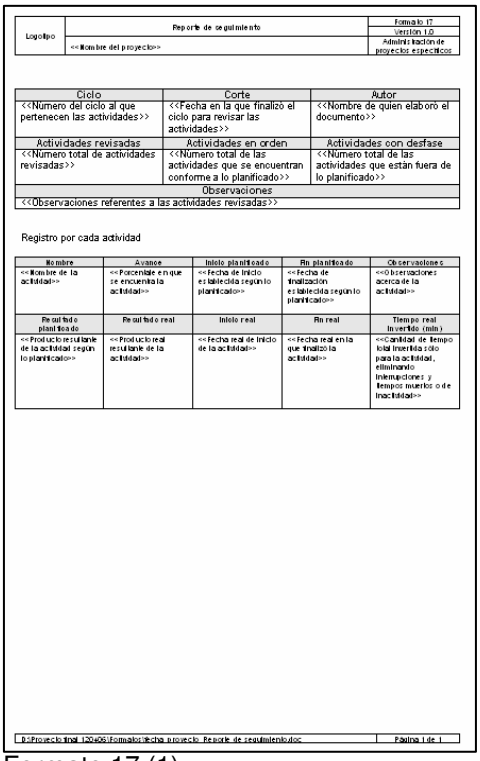


Logotipo	Minutas	Formato 16 Versión 1.0
	Proyecto: <<Nombre del proyecto>>	Administración de proyectos específicos

Responsable del proyecto	Lugar	Hora	Fecha
<<Nombre del líder de proyecto>>	<<Lugar de la reunión>>>	<<Hora de la reunión>>	<<Fecha de la reunión>>
Asistentes			
Nombre(s)	Área(s)	Puesto(s)	
<<Nombre del asistente>>	<<Nombre del área a la que pertenece el asistente>>	<<Nombre del puesto del asistente>>	
Agenda			
<<Temas a tratar en la reunión>>			
Aspectos relevantes			
<<Temas de importancia que fueron tratados en la reunión>>			
Fuera de alcance			
<<Temas que fueron tratados durante la reunión pero que no pertenecían a la agenda>>			
Acuerdos			
<<Acuerdos que se lograron entre los integrantes de la reunión>>			
Pendientes			
<<Temas que no pudieron abarcarse en la reunión>>			
Compromisos			
Actividad	Responsable	Fecha	
<<Descripción de la actividad comprometida a realizar>>	<<Nombre del responsable de ejecutar la actividad>>	<<Fecha para cumplir con la actividad>>	
Próxima reunión	<<Fecha acordada para la próxima reunión>>		
Documentos entregados	<<Nombre de los documentos entregados en la reunión>>		
Observaciones	<<Comentarios extra sobre la reunión>>		
Autor	<<Nombre de la persona que redactó la minuta>>		

D:\Formato.doc | Página 1 de 1

Formato 16 (1)



Logotipo	Reporte de seguimiento	Formato 17 Versión 1.0
	<<Nombre del proyecto>>	Administración de proyectos específicos

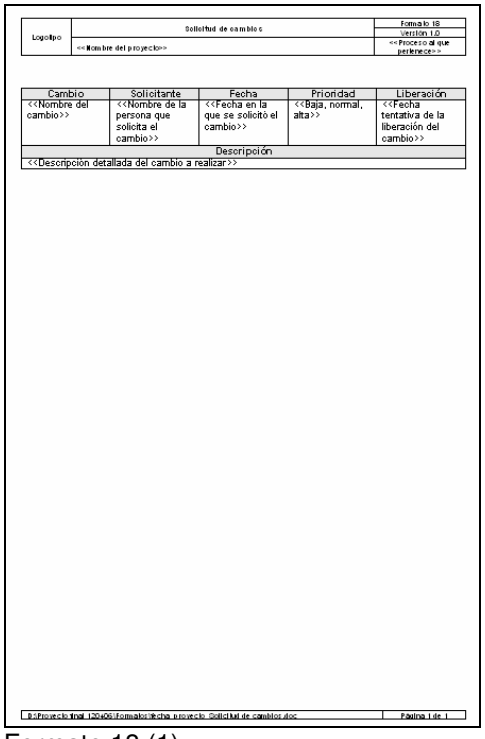
Ciclo	Corte	Autor
<<Número del ciclo al que pertenecen las actividades>>	<<Fecha en la que finalizó el ciclo para revisar las actividades>>	<<Nombre de quien elaboró el documento>>
Actividades revisadas	Actividades en orden	Actividades con desfase
<<Número total de actividades revisadas>>	<<Número total de las actividades que se encuentran conforme a lo planificado>>	<<Número total de las actividades que están fuera de lo planificado>>
Observaciones		
<<Observaciones referentes a las actividades revisadas>>		

Registro por cada actividad

Nombre	Avance	Inicio planificado	Fin planificado	Observaciones
<<Nombre de la actividad>>	<<Porcentaje en que se encuentra la actividad>>	<<Fecha de inicio establecida según lo planificado>>	<<Fecha de finalización establecida según lo planificado>>	<<Observaciones acerca de la actividad>>
Resultado planificado	Resultado real	Inicio real	Fin real	Tiempo real invertido (min)
<<Producto resultante de la actividad según lo planificado>>	<<Producto real resultante de la actividad>>	<<Fecha real de inicio de la actividad>>	<<Fecha real en la que finalizó la actividad>>	<<Cantidad de tiempo total invertido ciclo para la actividad, eliminando interrupciones y tiempos muertos o de inactividad>>

D:\Proyecto final 120406\Formatos\Ficha proyecto Reporte de seguimiento.doc | Página 1 de 1

Formato 17 (1)



Logotipo	Solicitud de cambios	Formato 18 Versión 1.0
	<<Nombre del proyecto>>	<<Proceso al que pertenece>>

Cambio	Solicitante	Fecha	Prioridad	Liberación
<<Nombre del cambio>>	<<Nombre de la persona que solicita el cambio>>	<<Fecha en la que se solicitó el cambio>>	<<Baja, normal, alta>>	<<Fecha tentativa de la liberación del cambio>>
Descripción				
<<Descripción detallada del cambio a realizar>>				

D:\Proyecto final 120406\Formatos\Ficha proyecto Solicitud de cambios.doc | Página 1 de 1

Formato 18 (1)

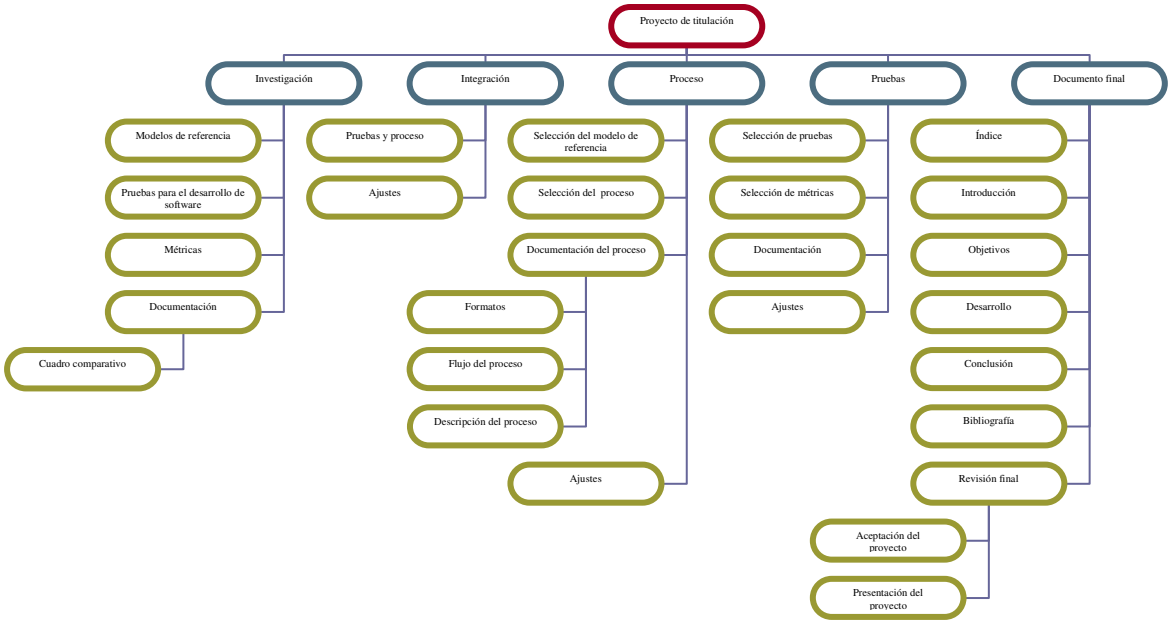
Documentos del proyecto

Mindmap

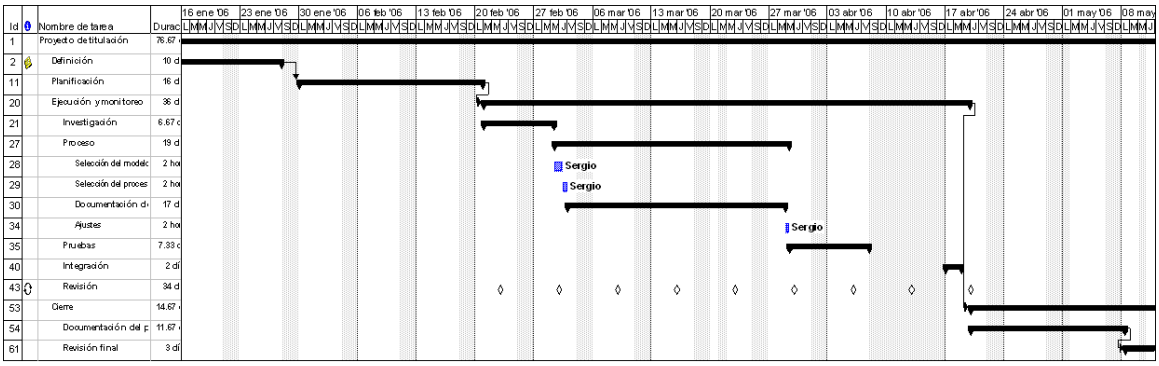
Work breakdown structure

El *work breakdown structure* es una descripción jerárquica del trabajo que debe ser realizado para completar el proyecto según la definición del POS.

Técnica utilizada para elaborar el WBS: *Mindmaps*



Calendario



Project Overview Statement

Problema/oportunidad
Mejorar el software que se desarrolla

Meta del proyecto
Contar con el proceso y las pruebas necesarias para el desarrollo del software

Objetivos del proyecto
Establecer un proceso para el desarrollo del software.
Establecer pruebas dentro del proceso que permitan asegurar que se está desarrollando el software correcto y de manera correcta.

Proporcionar una herramienta de apoyo dentro del proceso para la administración de los proyectos.

Criterios de éxito
El proyecto proporcionará las siguientes ventajas:

Al usuario y/o cliente
- Permitirá que los proyectos terminen en los tiempos establecidos
- Que el software se desarrolle dentro del presupuesto señalado
- Se podrá saber el avance del desarrollo, así como las fechas de entrega.
- Tener la seguridad de que se está desarrollando los que se ha pedido.
- Contar con software que funciona adecuadamente

Al equipo de desarrollo
- Saber que lo que se desarrolla está bien
- Contar con una forma ordenada de desarrollar software
- Establecer bases cuantitativas para mejorar el desarrollo del software

Suposiciones, riesgos y obstáculos
- Elaboración incorrecta del documento final.
- Tiempo insuficiente para terminar las actividades del proyecto
- Recursos insuficientes
- Ambigüedad del proyecto
- Información incompleta para elaborar el documento final
- Conocimientos insuficientes sobre los temas del proyecto
- Contar con bastante información para el proyecto
- Falta de retroalimentación

Riesgos

No.	Descripción del riego	Propietario del riesgo	Acción a tomar
1	Elaboración incorrecta del documento final.	Alumno, Asesor	Conocer las características requeridas para que sea aceptado el proyecto (letra, encabezados, citas, contenido, fechas de presentación, etc.).
2	Tiempo insuficiente para terminar las actividades del proyecto	Alumno	Elaborar un plan de trabajo que cubra todas las actividades del proyecto, así como imprevistos.
3	Recursos insuficientes	Alumno, Asesor	Contar con la infraestructura, referencias y material disponible para el proyecto.
4	Ambigüedad del proyecto	Alumno	Elaborar una definición los suficientemente clara de las pretensiones del proyecto.
5	Información incompleta para elaborar el documento final	Alumno	Documentar de forma paralela al desarrollo del proyecto conforme a los requerimientos necesarios del proyecto para que sea aceptado.
6	Conocimientos insuficientes sobre los temas del proyecto	Alumno, Asesor	Considerar que el material que se va a utilizar sea conocido, evitando lo más posible extender el tiempo en explicaciones de terceros.

No.	Descripción del riego	Propietario del riesgo	Acción a tomar
7	Contar con bastante información para el proyecto	Alumno	Seleccionar los elementos necesarios, excluyendo aquellos a los que sea necesario más tiempo o muy especializados.
8	Falta de retroalimentación	Asesor	Acordar y cumplir las citas acordadas para las asesorías.

Escala de riesgos durante todo el proyecto
1 = bajo riesgo
2 = riesgo medio
3 = alto riesgo

Cuadro de análisis de riesgos

		Riesgos								Total
		1	2	3	4	5	6	7	8	
Fases del proyecto	Definición	1	2	1	3	1	1	1	3	13
	Planificación	1	2	1	3	1	2	2	2	14
	Ejecución	1	3	3	2	1	3	3	2	18
	Cierre	3	3	3	1	3	1	3	3	20
Total		6	10	8	9	6	7	9	10	**65**

Nivel de riesgo para este proyecto 96/65 = 68%

Resource breakdown structure

Es una estructura organizada por actividades y en cada una de ellas está integrado por los recursos necesarios para realizar dicha actividad.

Recursos humanos

Perfil: investigador
Habilidades:

- Indagar las fuentes de información disponibles
- Recolectar información afín al proyecto

Perfil: Documentador
Habilidades:

- Recolectar la información generada por el proyecto
- Manejo en el software comúnmente utilizado en documentación
- Claridad y orden en la presentación del documento
- Uso correcto de las reglas de redacción y ortografía

Perfil: Experto en calidad del software
Habilidades:

- Experiencia en las diferentes actividades de la calidad del software
- Conocimiento de estándares, modelos de referencia, libros, revistas especializadas, etc.

Perfil: Experto en procesos
Habilidades:

- Experiencia en el diseño e implementación de procesos

Perfil: Experto en pruebas y métricas de software
Habilidades:

- Experiencia en la aplicación de pruebas y métricas en el desarrollo de software
- Conocimiento de estándares y modelos de referencia

Perfil: Asesor
Habilidades:

- Experiencia en la dirección de proyectos de titulación académicos

Perfil: Corrector de estilos
Habilidades:

- Experiencia en la corrección de textos de titulación académicos

Analista de información
Habilidades:

- Análisis crítico de la información recolectada
- Selección y simplificación de la información útil al proyecto
- Discernimiento de las condiciones dadas del proyecto con respecto a la información proporcionada

Equipo

Requerimientos mínimos

Computadora:

- Procesador 750 Mhz
- Memoria RAM 256 Mb
- MODEM o tarjeta de red
- Conexión a Internet
- Disco duro 10 Gb
- Accesorios: teclado, ratón, monitor, puerto USB, unidad de quemador de CD, puerto para impresora

Impresora:

- Inyección de tinta
- Impresiones en blanco y negro y a color
- Conexión a puerto paralelo o USB
- Controladores de instalación

Software:

- MS Office 2003
- Windows XP
- MS Project 98
- Acrobat reader 6.0

- MS Visio 2003
- Internet Explorer 6.0

Fuentes de información

- Sitios de Internet especializados
- Revistas especializadas
- Foros
- Especialistas en el tema
- Organizaciones
- Bibliotecas
- Libros especializados en procesos de software
- Libros especializados en pruebas de software
- Libros especializados en métricas de software
- Fuentes de consulta complementaria

Glosario

Actividad	- Conjunto de actos o labores específicas a realizar por un individuo, departamento o unidad. - Conjunto de operaciones afines y sucesivas que desarrolla una misma persona o una misma unidad. - Conjunto de operaciones. - Un conjunto de acciones diseñadas para alcanzar un resultado en particular. Las actividades son usualmente definidas como parte de un proceso o planes y son documentadas en procedimientos. (ITIL)
Administración	- Actividad humana que apoyada en diferentes técnicas, busca optimizar la integración de los recursos dentro de una entidad para el logro de sus objetivos.
Administración de riesgos	- La administración de riesgos desarrolla una disciplina y un ambiente de decisiones y acciones proactivas para valorar ininterrumpidamente lo que puede fallar, determinar cuáles son los riesgos importantes que se deben de enfrentar e implementar estrategias para abordarlos.
Administrar	- Hacer a través de los demás.
Aplicación	- Software que proporciona funciones que son requeridas por los servicios de IT. Cada aplicación puede ser parte de uno o más servicios de IT. Una aplicación corre en uno o más servidores o clientes. (ITIL)
Aseguramiento	- Actividad que obtiene el acuerdo de la gerencia de que un proceso, plan u otro entregable se encuentra completo, exacto, confiable y se encuentra acorde a los requerimientos especificados. Aseguramiento es diferente a la auditoría, la cual es más enfocada con el cumplimiento de estándares formales. (ITIL)
Aseguramiento de la calidad	- Un conjunto de actividades planificadas y sistematizadas necesarias para proporcionar una adecuada certeza de que el producto o servicio se encuentra libre de defectos. - Una función *staff*, creada para apoyar la implementación de la administración de la calidad total.
Atributo	- Una propiedad mensurable, física o abstracta, de una de una entidad (ISO 14598-1:1999). Un atributo se puede medir (cuantificar) por medio de una métrica directa o indirecta.
Calidad	- La calidad de un producto es ampliamente determinado por la calidad del proceso que es usado para desarrollarlo y mantenerlo. - Propiedad o conjunto de propiedades inherentes en algo, que permiten juzgar su valor. - Todas las características de una entidad que contienen la capacidad de satisfacer necesidades expresadas e implícitas. - Capacidad de un producto, servicio o proceso para proporcionar el valor pretendido. (ITIL)
Calidad del software	- Concordancia del software producido con los requisitos explícitamente establecidos, con los estándares de desarrollo expresamente fijados y con los requisitos implícitos, no establecidos formalmente, que desea el usuario (R. Pressman).

Capacidad del proceso de área	- Es un conjunto de procesos relacionado con una simple área de proceso o práctica específica.
Capacidad del proceso de software	- Rango de los resultados esperados que pueden ser realizados por el seguimiento de un proceso de software.
Casos de prueba.	- Diseño de pruebas que permiten encontrar el mayor número de errores con la mínima cantidad de esfuerzo y tiempo posible (R. Pressman). - Una condición que se probará que incluye su propia identificación y la respuesta esperada. (Marnie L. Hutcheson). - La especificación detallada en cada caso de uso proveniente de la especificación del diseño de pruebas (IEEE 829). - Estructura que permita probar todos los procesos posibles del sistema para encontrar sus inadecuaciones con los requerimientos y normas establecidas, con el menor esfuerzo y tiempo posibles. - Un conjunto de entradas, condiciones de ejecución y resultados esperados desarrollados para un objetivo particular como, por ejemplo, ejercitar un camino concreto de un programa o verificar el cumplimiento de un determinado requisito.
Categorías de pruebas	- Agrupación conceptual varias pruebas bajo alguna característica en común.
Ciclo de vida	- Una aproximación lógica a la adquisición, el suministro, el desarrollo, la explotación y el mantenimiento de software (IEEE 1074) - Un marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de software, abarcando la vida del sistema desde la definición de los requisitos hasta la finalización de su uso. (ISO 12207-1)
Confiabilidad	- Es una de las medidas para evaluar cuantitativamente la calidad de un producto de software. - La probabilidad de que el software opere libre de fallas en un periodo de tiempo y ambiente específico.
Control	- Proceso que se encarga de eliminar el caos y proporciona congruencia a la organización con el propósito de que pueda alcanzar sus objetivos. - Proceso que permite garantizar que las actividades reales se ajusten a las actividades proyectadas en la planificación. - Administrar un riesgo o asegurar de que los objetivos del negocio sean alcanzados. (ITIL)
Control de calidad	- Proceso por el cual el producto correcto es determinado y las acciones son iniciadas cuando una inconformidad es detectada. - Función lineal; trabajo realizado dentro de un proceso para asegurar que el producto del trabajo se encuentra de acuerdo con los estándares y requisitos.
Coordinar	- Enlazar diferentes unidades de la organización.
Costo	- La suma de dinero que se gasta en una actividad en específico, Servicio de IT o unidad de negocio. El costo consiste en costro real (dinero), costo teórico como tiempo de las personas y depreciación. Costo es también usado con el nombre de política de cargo que recupera el costo exacto de proveer un servicio. (ITIL)

Dato	- Registro de los hechos, acontecimientos, etc. - Una abstracción de la realidad.
Defecto	- Variación del atributo de un producto deseado. Algunos defectos pueden llegar a ser una falla. Un defecto puede causar muchas fallas. Muchos defectos son causados por procesos que no trabajan apropiadamente. - Cualquier desviación de las especificaciones; equivocado, perdido o ausente, o extra (productor). - Cualquiera que cause insatisfacción del usuario (usuario). - Un proceso, una definición de datos o un paso de procesamiento incorrectos en un programa. - Sinónimo de error. (ITIL)
Departamentalización	- Dividir y agrupar las funciones y actividades que se deberán llevar a cabo por un grupo de personas.
Depuración	- El proceso de localizar, analizar y corregir los defectos que se sospecha que contiene el software.
Diagrama	- Representación gráfica de lo que se va a hacer, quién lo va a hacer, cuándo se va a hacer, dónde se va a hacer y para qué se va a hacer, la sucesión con que se realizan las operaciones de un procedimiento y/o recorrido de formas o materiales, muestra las unidades administrativas o los puestos que intervienen en cada operación, por medio de símbolos convencionales.
División del trabajo	- Es la acción de separar las actividades de un puesto o de una unidad administrativa con el fin de llevar a cabo una función con el mayor cuidado, precisión y eficiencia y con el mismo esfuerzo.
Documento	- Es cuando una forma elaborada se ha completado o llenado con los datos solicitados. - Información que puede ser leída. Un documento puede ser en papel o de manera electrónica. (ITIL)
Entidad	- Un objeto que va a ser caracterizado mediante una medición de sus atributos (ISO-15939).
Error	- Acción desacertada o equivocada. - La diferencia entre un valor calculado, observado o medido y el valor verdadero, especificado o teóricamente correcto. Por ejemplo, una diferencia de dos centímetros entre el valor calculado y el real. - Una acción humana que conduce a un resultado incorrecto (*mistake*). Por ejemplo, en el operador o el programador pulse una tecla equivocada.
Especificación	- Definición formal de requerimientos. Una especificación puede se utilizada para definir requerimientos técnicos u operativos, los que pueden ser internos o externos. Muchos estándares públicos consisten en códigos de prácticas y una especificación. La especificación define el estándar contra la cual una organización puede ser auditada. (ITIL)
Especificación del diseño de prueba	- Identifica el conjunto de características que serán probadas y describe el grupo de casos de prueba a la que serán sometidas esas características. En suma, es el refinamiento para que se alcance lo que se encuentra en el plan de pruebas (IEEE 829). - Identificación de características encontradas en un plan de pruebas que serán asignadas a los casos de prueba correspondientes.

Especificación del proceso de prueba	- Especifica los pasos para ejecutar los casos de prueba y el proceso para determinar si el proceso pasó o falló la prueba (IEEE 829). - Criterio para realizar y evaluar los resultados del los casos de prueba.
Estado	- Representa un modo externo de comportamiento.
Estándar	- Representación abstracta de un producto que define el nivel mínimo de rendimiento, robustez, organización, etc., que debe alcanzar el producto desarrollado. - Un requerimiento obligatorio. (ITIL)
Estimación	- El uso de la experiencia para proveer un valor aproximado para una métrica o costo. (ITIL)
Estrategia	- Plan general. - Panorama general en el que se sabe claramente la dirección que se va a tomar.
Estudio del trabajo	- Ciertas técnicas que se utilizan para examinar el contenido humano en todos sus contextos. Permite investigar todos los factores que influyen en la eficiencia y economía de la situación estudiada con el fin de obtener y efectuar mejoras.
Falla	- Un defecto que causa un error en la operación o impacta negativamente a un usuario. - La incapacidad de un sistema o de alguno de sus componentes para realizar las funciones requeridas dentro de los requisitos de rendimiento especificados. - Pérdida de la disponibilidad para operar una especificación o para entregar un producto requerido. El término puede ser usado cuando se refiere a servicios IT, procesos, actividades, elementos de configuración, etc. Una falla con frecuencia causa un incidente. (ITIL)
Forma	- Es la herramienta o medio de comunicación escrito que por lo general contiene información fija escrita y un espacio para información variable.
Función	- Conjunto de actividades que se tienen asignados. - Conjunto de procedimientos asignados a un área. - Conjunto de actividades afines y coordinadas, necesarias para alcanzar los objetivos de un organismo social. - Un propósito previsto de un elemento de configuración, persona, equipo, proceso o servicio IT. (ITIL)
Función sustantiva	- Es la que identifica la esencia de la dependencia o unidad administrativa y que desarrollan sus órganos para el cumplimiento de objetivos en forma directa.
Herramienta	- Es el vehiculo para desempeñar un proceso de pruebas. (William Perry)
Implantar	- Establecer y poner en ejecución nuevas doctrinas, instituciones, prácticas o costumbres.
Implementar	- Poner en funcionamiento, aplicar métodos, medidas, etc., para llevar algo a cabo.
Información	- Resultado del procesamiento de los datos con el fin de que sean útiles y significativos.
Informática	- Término en frances *informatique* fue creado por la academia francesa en 1966, el acrónimo está conformado por las palabras *information* y *automatique*. Definiendo informática como la ciencia del tratamiento racional, principalmente a través de las máquinas automáticas de la información, entendida como la base de los conocimientos humanos.

Inspección	- Técnica de evaluación formal en la cual los requisitos de software, diseño o la codificación se examinan en detalle por una persona o grupo distinto del autor, para detectar defectos, disconformidades con las normas de desarrollo y otros problemas (IEEE 1990). - Uno o más ingenieros revisan los productos de otro ingeniero para encontrar en ellos defectos y problemas. (TSPi)
IT	- *Information Technology*
ITIL	- IT *Infraestructure library* es una guía de buenas prácticas para la administración de servicios de IT. Consiste en una serie de publicaciones que proporcionan una guía en provisión de calidad de los servicios y en los procesos y requerimientos para el soporte.
KPAs (*Key process Area*)	- Es una agrupación de procesos o prácticas clave. - Identifican un grupo de actividades relacionadas que cuando son ejecutadas en forma colectiva, alcanzan un conjunto de propósitos considerados importantes para mejorar el potencial de los procesos. - Identifica los tópicos que deben ser tomados en cuenta para lograr un nivel de madurez.
Línea base	- Registro de un estado en un punto específico en el tiempo. Puede ser creado para una configuración, un proceso o cualquier otro tipo de datos. (ITIL)
Madurez del proceso de software	- Alcance para el cual un proceso específico está explícitamente definido, dirigido, medido, controlado y efectuado.
Madurez organizacional	- Un conjunto de áreas de proceso a través de una organización.
Medible	- Es una relación abstracta entre atributos de una o más entidades, y una necesidad de información (ISO 15939).
Mejores prácticas	- Actividad o proceso que ha sido exitosamente usado por múltiples organizaciones. ITIL es un ejemplo de mejores prácticas. (ITIL)
Método	- Se identifica como la manera de efectuar una operación o una secuencia de actividades ordenada de manera cronológica y eficiente de operaciones para obtener un resultado.
Metodología	- Conjunto de filosofías, fases, procedimientos, reglas, técnicas, herramientas, documentación y aspectos de formación para los desarrolladores de sistemas de información. - Un conjunto de procedimientos, técnicas, herramientas, y un soporte documental que ayuda a los desarrolladores a realizar nuevo software.
Métrica	- Es algo que es medido y reportado para ayudar a administrar un proceso, servicio de IT o una actividad. (ITIL)
Métrica de software	- Es una medición de alguna de las propiedades de una parte del software o sus especificaciones.
Misión	- Objetivo fundamental y razón de ser. - Es una descripción corta del propósito general e intensiones de una organización. Es el lo que se quiere alcanzar, pero no cómo debería alcanzarse. (ITIL)
Modelo	- Es una colección de elementos estructurados que describen características de un proceso efectivo.

	- Representación de la realidad.
Monitoreo	- Indica el progreso de un proyecto.
Norma	- Regla que se debe seguir o ajustar las conductas, tareas, actividades, etc.
Notas liberadas	- Especifica los puntos que han sido sometidos a la prueba (IEEE 829).
Objetivo	- Es el punto al que se quiere llegar. - Propósito definido de un proceso, una actividad o una organización como un todo. Objetivos son usualmente expresados como objetivos que pueden medirse. El término objetivo es también utilizado de manera informal para referirse a un requerimiento. (ITIL)
Operación	- Mínima expresión de trabajo. - Es la división mínima del trabajo. - Pueden ser físicas o mentales; en conjunto conforman una actividad.
Organigrama	- Representación gráfica de la estructura de una organización. - Es una representación gráfica de la estructura orgánica de una institución o de una de sus áreas o unidades administrativas, en las que se muestran las relaciones que guardan entre sí los órganos que la componen.
Órgano	- Es el tipo de unidad administrativa impersonal que conforma una organización, facultado para ejercer funciones de autoridad para toma de decisiones; ejemplo: Gerencia, subgerencia, departamento, etc.
Plan	- Define las actividades del proyecto. - Documento que identifica una serie de actividades y los recursos requeridos para alcanzar un objetivo. (ITIL)
Plan de prueba	- Describen el alcance, acercamiento, recursos y programación de las actividades de prueba (IEEE 829). - Documento *a priori* que contiene las actividades, recursos, alcance y tiempos para un sistema dependiendo de sus características y circunstancias.
Política	- Reglas que enfocan el objetivo de la empresa, planes de acción. - Lineamientos generales de acción, son flexibles y no tienen sanción. - Son normas de carácter general que guían la actuación de las dependencias y sus integrantes acerca de las funciones sustantivas y las relaciones de mando de acuerdo con los objetivos de los procesos a su cargo. - Son guías básicas que sirven como marco legal de actuación para la realización de acciones en una organización. - Son las intenciones y expectativas de administración formalmente documentadas. Son utilizadas para decisiones directas y para asegurar un desarrollo e implementación consistente y apropiado de procesos, estándares, roles, actividades, infraestructura, etc. (ITIL)
Presupuesto	- Es un programa con valor monetario y en función de tiempo y trabajo.
Procedimiento	- Es el que indica cómo hacer algo con una serie de pasos que se hacen de manera repetida. - Conjunto de actividades para realizar una función.

	- Sucesión cronológica y secuencial de operaciones concatenadas entre sí, que se constituyen en una unidad, en función de la realización de una actividad o tarea específica dentro de un ámbito predeterminado de aplicación. - Todo procedimiento involucra actividades y tareas del personal, la determinación de tiempos de realización, el uso de recursos materiales y tecnológicos y la aplicación de métodos de trabajo. - Un procedimiento es una serie de actividades u operaciones ligadas entre sí por un conjunto de empleados, ya sea dentro de un mismo departamento o abarcando varias direcciones de una empresa. - Documento que contiene los pasos que especifican cómo alcanzar una actividad. Los procedimientos son definidos como parte de los procesos. (ITIL)
Proceso	- Un conjunto de prácticas desempeñadas para alcanzar un propósito dado; esto incluye métodos, herramientas, materiales y/o gente. (CMMI) - Secuencia de pasos realizados para conseguir un propósito. - Conjunto de procedimientos o subsistemas. - Es un conjunto estructurado de actividades diseñadas para cumplir con un objetivo específico. Un proceso toma una o más entradas definidas y las convierte en salidas definidas. Un proceso puede incluir cualquiera de los roles, responsabilidades, herramientas y controles administrativos requeridos para entregar salidas confiables. Un proceso puede definir políticas, estándares, guías, actividades e instrucciones de trabajo si es que son necesarios. (ITIL)
Proceso software	- Conjunto de actividades, métodos, prácticas y transformaciones que la gente utiliza para desarrollar y mantener software.
Programa	- Pasos principales para llegar a un objetivo. - Un número de proyectos que son planificados y administrados juntos para alcanzar un objetivo general. (ITIL)
Proyecto	- Búsqueda de la solución a un problema o necesidad tomando en cuenta costos y tiempo. - Una organización temporal con gente y otros recursos requeridos para alcanzar un objetivo. Cada proyecto tiene un ciclo de vida que incluye inicio, planificación, ejecución, cierre, etc. Los proyectos son usualmente administrados utilizando una metodología formal como Prince2. (ITIL)
Prueba	- Comprueba la respuesta de un sistema a los estímulos y compara esa respuesta a un estándar. Evalúa la calidad de la respuesta con respecto al estándar. Proporcionando un software y una lista de funciones que se supone que desempeñará, buscando que desempeñe esas funciones como son descritas. Adicionalmente, se buscarán cosas que no han sido descritas (Marnie L. Hutcheson). - Una actividad en la cual un sistema o uno de sus componentes se ejecuta en circunstancias previamente especificadas, los resultados se observan y registran y se realiza una evaluación de algún aspecto. - Proceso de ejecutar un programa con el fin de encontrar errores (Myers).

	- El conjunto casos y procedimientos de prueba (IEEE 1990). - Observación y registro del comportamiento de un software en específico que posteriormente se cotejará a un sistema de valoración, siendo esto los objetivos, especificaciones, requerimientos y expectativas. Con esto se logrará identificar su idoneidad para circunstancias específicas. - Permiten verificar y validar el software cuando ya está en forma de código ejecutable (M. Piattini). - Sólo involucra el componente *check* del ciclo *plan-do-check-act* (PDCA). (Perry, William) - El propósito es evaluar el producto y no arreglarlo. (TSPi)
Prueba dinámica	- Verificación o validación realizada que ejecuta el código del sistema.
Prueba estática	- Verificación realizada sin ejecutar el código del sistema.
Pruebas de caja blanca	- Pruebas basadas en el conocimiento de la estructura interna del código y su lógica.
Pruebas de caja negra	- Pruebas basadas en especificaciones externas sin conocimiento de como se encuentra estructurado el sistema.
Pruebas estructurales	- Pruebas que validan la arquitectura del sistema.
Pruebas funcionales	- Son pruebas de requisitos del negocio (lo que el sistema supuestamente hace).
Puesto	- La plaza necesaria y permanente para es desarrollo normal de la organización.
Realización del proceso del software	- Resultados alcanzados por el seguimiento de un proceso de software.
Registro de pruebas	- Proporciona un registro cronológico acerca de detalles relevantes observados durante la ejecución de la prueba. (IEEE 829).
Regla	- Guía de acción general que puede tener una sanción.
Reporte de incidente de la prueba	- Documenta cualquier evento observado durante la prueba que requiere mayor investigación (IEEE 829).
Reporten del resumen de la prueba	- Reúne los resultados de las actividades de prueba y proveer una evaluación basada en esos resultados (IEEE 829).
Requerimiento	- Sentencia formal de lo que se necesita. (ITIL)
Revisión	- Comprobar los productos realizados por otra persona para encontrar defectos.
Riesgo	- Es la posibilidad de sufrir una pérdida.
Rol	- Un conjunto de responsabilidades definidas en un proceso y asignados a una persona o equipo. Una persona o equipo puede tener múltiples roles.
Servicio	- Proporcionar algo de valor a un cliente que no son productos (cosas físicas con valor material). (ITIL)
Simplificación del trabajo	- El uso organizado del sentido común para encontrar formas más fáciles y mejores de realizar una tarea. La simplificación del trabajo posibilita la racionalización de las tareas, lo que permite el abatimiento de los costos, menor inversión de capital y el mejoramiento de la rentabilidad de los recursos.
Sistema	- Conjunto ordenado de procedimientos (operaciones y métodos), relacionados entre sí, que constituyen a realizar una función. - Una red de procedimientos relacionados entre sí y desarrollados de acuerdo a un esquema integrado para lograr una mejor actividad de la empresa. - Complejo de elementos interactuantes. (Bertalanffy, Ludwig Von, *Teoría general de los sistemas*. P. 56)

	- Un número de cosas relacionadas que trabajan juntas para alcanzar los objetivos. (ITIL)
Software	- Un programa de computadora que desempeña un conjunto de funciones. - Son los programas de ordenador, los procedimientos y, posiblemente, la documentación asociada y los datos relativos a la operación del sistema informático (IEEE 1990).
Supervisión	- Revisar y comparar los logros y los resultados obtenidos frente a las estimaciones, los compromisos y los planes del proyecto, actualizándolos en función de estos resultados.
Táctica	- Nivel intermedio de los tres niveles de planificación y entrega (estratégico, táctico, operativo). Las actividades tácticas incluyen planes a mediano plazo requeridos para alcanzar objetivos específicos, típicamente en periodos de semanas y meses. (ITIL)
Tarea	- Trabajo que debe hacerse en tiempo limitado. - Actividad con un comienzo y un fin.
Técnica	- Método o procedimiento (con referencia a detalles prácticos o formales), o forma de usar habilidades básicas, en la representación de un trabajo artístico o realizando una operación mecánica o científica (Marnie L. Hutcheson). - Especialización en un tema en una perspectiva teórico-práctica. - Es el proceso para asegurar que algunos aspectos de la aplicación funcionan apropiadamente. (William Perry)
Transición	- Obliga al paso de un estado a otro si se cumple la condición
Validación	- Actividad de garantiza que el producto final cumple con las especificaciones. - El proceso de evaluación de un sistema o de uno de sus componentes durante o al final del proceso de desarrollo para determinar si satisface los requisitos especificados (IEEE 1990). - ¿Estamos construyendo el producto correcto? (Boehm).
Verificación	- Actividad que asegura provisionalmente que los entregables del producto cumplen con las especificaciones de entrada. - El proceso de evaluación de un sistema o de uno de sus componentes para determinar si los productos de una fase dada satisfacen las condiciones impuestas al comienzo de dicha fase (IEEE 1990). - ¿Estamos construyendo correctamente el producto? (Boehm).
Visión	- Perspectiva a futuro. - Es una descripción de lo que una organización quiere llegar a ser en el futuro. Una visión es utilizada de apoyo en la cultura y en el plan estratégico. (ITIL)

Bibliografía

Beizer, Boris. *Software Testing Techniques*. New York, Van Nostrand Reinhold, 1983.

Bertalanffy, Ludwig Von. *Teoría general de los sistemas.* México, FCE, 2003.

Chrissis, Mary Beth, et. al. *CMMI Guidelines for process integration and product improvement*. México, Addison-Wesley, 2003.

Copeland, Lee. *A Practitioner's Guide to Software Test Design.* Boston, Artech House Publishers, 2004.

Fendon, Norman e. y Pfleeger, Shari Lawrence. *Software metrics*. Boston, Thomson, 1997.

Frondizi, Risieri. *¿Qué son los valores?* México, FCE, 2004.

Humphrey, Watts S. *A discipline for software engineering*. Massachusetts, Addison-Wesley, 1995.

Hutcheson, Marnie L. *Software Testing Fundamentals: Methods and Metrics*. Indianapolis, Wiley, 2003.

Jorgensen, Paul C. *Software Testing: A Craftsman's Approach*. Boca Raton, Florida, CRC, 2002.

Myers, Glenford J. *El arte de probar el software*. Trad. Clelia Chamatropulos de Filevich y Alberto Filevich. Buenos aires, México, El ateneo, 1983.

Nicol, Eduardo. *La idea del hombre*. México, FCE, 2003.

Oyvind, Mo. *Comparación del modelo de procesos para la industria de software (MoProSoft) con las normas y modelos de referencia*. Tesis para obtener el grado de maestro en ciencias e ingeniería en computación. México, UNAM, 2005.

Perry, William E. *Effective methods for software testing*. New York, Wiley, 1995.

Pfleeger, Shari Lawrence. *Software Engineering: Theory and Practice*. New Jersey, Prentice Hall, 2001.

Piattini, Mario G. *Análisis y diseño detallado de aplicaciones informáticas de gestión*. Madrid, RA-MA, 1996.

Price, Jonathan. *How to write a computer manual: A handbook of software documentation*. California, Benjamin/Cummings, 1984.

Wang Y., et. al.”Towards Software Process Excellence: A survey report on the best practices in the software industry” en *ASQ Journal of Software Quality Professional*, Vol. 2, No. 1, pp.34-43 (1999).

Whittaker, James A. *How to break software*. Boston, Pearson Education, 2003.

Estándares

IEEE STD 829 – 1998

ISO/IEC 9126 – 1 2000

ISO/IEC 9126 – 2 2002

ISO/IEC 9126 – 3 2002

ISO/IEC 9126 – 4 2001

MoProSoft versión 1.3 (coloreado) 2005

Project Management Institute. *Project management Body of Knowledge*. Pennsylvania, PMI, 2004. [3era ed.]

Software Engineering Laboratory. *Software measurement guidebook*. Greenbelt, National Aeronautics and Space Administration, 1995.

SWEBOK 2004

Páginas de referencia

International Software Testing Institute
http://www.softtest.org/

Portal Oficial de la Industria Mexicana de Software
http://www.software.net.mx

Asociación Mexicana para la Calidad en la Ingeniería de Software
http://www.amcis.org.mx

Carnegie Mellon. Software Engineering Institute
http://www.sei.cmu.edu