3. Deseño e realización de probas

3.1. Introdución ás probas de software

3.1.1 Conceptos

Calidade do software

A **calidade do software** é o grado no que o software cumpre cos **requisitos funcionais** establecidos co cliente, cos **estándares de desenvolvemento** explicitamente documentados e coas características que se esperan de calquera software desenvolvido profesionalmente. Desta definición pode deducirse que:

- Se o software non cumpre cos requisitos iniciais non será de calidade.
- Os estándares de desenvolvemento guían na forma de elaborar o software polo que, se o software non cumpre con eses estándares, non será de calidade.
- Se o software ten erros importantes, ou non é fácil de utilizar ou é difícil de manter, non será de calidade.

Determinar a calidade é moi complicado debido á natureza do software xa que non é un elemento tanxible como outros produtos industriais, pero debe estar libre de erros, poden facerse medicións sobre el e pódese validar o proceso de desenvolvemento do mesmo.

En todo proceso de desenvolvemento de software realízanse controis periódicos, normalmente coincidindo cos fitos do proxecto que deben permitir:

- **Verificar** que se está a construír correctamente o produto.
- **Validar** que se está a construír o produto correcto, é dicir, o que realmente quere o usuario.

Estándares e certificacións

A normalización ou estandarización é o proceso de elaborar, difundir, aplicar e mellorar as regras que se utilizan en distintas actividades científicas, industriais ou económicas co fin de ordenalas e melloralas. Segundo a ISO (*International Organization for Standardization*), a **normalización** é a actividade que ten por obxecto establecer, ante problemas reais ou potenciais, disposicións destinadas a usos comúns e repetidos, co fin de obter un nivel de ordenamento óptimo nun contexto dado, que pode ser tecnolóxico, político ou económico. A normalización permite:

- Simplificar, é dicir, reducir os modelos para quedar só cos máis necesarios.
- Unificar para permitir o intercambio a nivel internacional.
- Especificar para evitar erros de identificación creando unha linguaxe clara e precisa.

A **certificación** é a acción levada a cabo por unha entidade recoñecida, independente das partes interesadas, mediante a que se manifesta que unha organización, produto, proceso ou servizo cumpre os requisitos mínimos establecidos nunhas normas ou especificacións técnicas. Leva á empresa a diferenciarse do resto e tomar vantaxe no mercado, ao demostrar que segue uns estándares de calidade. A petición dunha certificación ten carácter voluntario, carrexa uns custos e ten que renovarse periodicamente.

ISO

A Organización Internacional de Normalización ou **ISO** (http://www.iso.org) naceu o 23 de febreiro de 1947 e é un organismo non gobernamental encargado de promover o desenvolvemento de normas internacionais de fabricación, comercio e comunicación para todas as ramas industriais fóra da eléctrica e a electrónica. A súa función principal é a de buscar a estandarización de normas de produtos e seguridade para as empresas ou organizacións a nivel internacional. As normas que produce denomínanse normas ISO. A central está en Xenebra (Suíza) pero está integrada polos organismos de normalización nacionais de moitos países. O contido dos estándares está protexido por dereitos de copyright e para acceder a eles o público corrente debe comprar cada documento na súa páxina oficial.

ISO 25000 tamén coñecida como SQuaRE (System and Software Quality Requirements and Evaluation) é unha familia de normas que ten por obxectivo a creación dun marco de traballo común para avaliar a calidade dun produto software.

A certificación da calidade do produto software con ISO 25000 permite ás empresas que desenvolven software coñecer a calidade dos seus produtos e ás empresas que compran software, decidirse por unha solución ou outra en función das súas necesidades.

A ISO 25000 está articulada en varias divisións entre as que podemos destacar:

- **ISO 25040** que define o proceso de avaliación da calidade do produto software.
- **ISO 25010** que determina as características e subcaracterísticas de calidade que se poden avaliar para un produto software.

Entre os motivos máis destacados polos que unha organización pode interesarse por avaliar o seu produto segundo a ISO 25000 podemos incluír:

- Diferenciarse dos seus competidores asegurando tempos de entrega e redución de fallos no produto tras a implantación en produción.
- Establecer acordos no ámbito do servizo, definindo determinados parámetros de calidade que o produto debe cumprir antes de entregarse.

- Detectar os defectos no produto software e proceder á súa eliminación antes da entrega e, polo tanto, aforrando custes na fase de mantemento posterior.
- Avaliar e controlar o rendemento do produto software desenvolvido, asegurando que poderá xerar resultados tendo en conta restricións de tempo e recursos establecidos.
- Asegurar que o produto software desenvolvido respecta os niveis necesarios para as características de seguridade.
- Comprobar que o produto desenvolvido poderá poñerse en produción sen poñer en compromiso o resto de sistemas e mantendo a compatibilidade coas interfaces necesarias.

IEC

A Comisión Electrotécnica Internacional ou CEI (http://www.iec.ch/) ou IEC (*International Electrotechnical Commission*) é unha organización de normalización nos campos eléctrico, electrónico e de tecnoloxías relacionadas. Foi fundada en 1904 durante o Congreso Eléctrico Internacional de San Luís (EEUU). Actualmente a súa sede está en Xenebra (Suíza) e está integrada polos organismos nacionais de normalización, nas áreas indicadas, dos países membros. En 1938, o organismo publicou o primeiro dicionario internacional *(International Electrotechnical Vocabulary*) co propósito de unificar a terminoloxía eléctrica, esforzo que se mantivo durante o transcurso do tempo, sendo o Vocabulario Electrotécnico Internacional un importante referente para as empresas do sector.

Numerosas normas desenvólvense conxuntamente coa ISO e chámanse normas ISO/IEC. Por exemplo, a ISO/IEC 25000.

AENOR

A Asociación Española de Normalización e Certificación ou AENOR é a única entidade recoñecida en España para desenvolver tarefas de normalización e certificación e para representar a España nos organismos europeos (CEN, CENELEC e ETSI) e internacionais (ISO e IEC). É unha entidade privada sen fins lucrativos que se creou en 1986. No seu sitio oficial (http://www.aenor.es) ofrece información sobre o proceso de certificación, publicación de novas normas, xornadas....

As normas que adapta ou elabora, coñecidas como normas UNE, indican como debe ser un produto ou como debe funcionar un servizo para que sexa seguro e responda ao que o consumidor espera del.

Os certificados AENOR son moi valorados, non só en España senón tamén no ámbito internacional, emitindo certificados en máis de 60 países. AENOR sitúase entre as 10 certificadoras máis importantes do mundo.

IEEE

IEEE (lido i-e-cubo en España e i-triplo-e en latinoamérica) corresponde ás siglas de *Institute of Electrical and Electronics Engineers*, en español Instituto de Enxeñeiros Eléctricos e Electrónicos, e é unha asociación técnico profesional mundial dedicada á estandarización, entre outras cousas. Naceu en 1884 pero adoptou o nome de IEEE en 1963 e na actualidade é a maior asociación internacional sen ánimo de lucro formada por profesionais das novas tecnoloxías, como enxeñeiros eléctricos, enxeñeiros en electrónica, científicos da computación, enxeñeiros en informática, enxeñeiros en biomédica, enxeñeiros en telecomunicación e enxeñeiros en mecatrónica. Segundo o mesmo IEEE, o seu traballo é promover a creatividade, o desenvolvemento e a integración, compartir e aplicar os avances nas tecnoloxías da información, electrónica e ciencias en xeral para beneficio da humanidade e dos mesmos profesionais. Colaboran con ISO e IEC en temas comúns. O sitio oficial é: http://www.ieee.org. A Sección Española do IEEE é recoñecida dentro da Rexión 8 en abril de 1968 e o seu sitio oficial é: http://www.ieeespain.org/.

Algunhas normas relacionadas co software son:

- IEEE 730. Plans de aseguramento da calidade de software.
- IEEE 829. Documentación de probas do software.
- IEEE 982.1, 982.2. Dicionario estándar de medidas para producir software fiable.
- IEEE 1008. Probas unitarias de software.
- IEEE 1012. Verificación e validación de software.
- IEEE 1028. Revisións de software.
- IEEE 1044. Clasificación estándar para anomalías do software.
- IEEE 1061. Estándar para unha metodoloxía de métricas de calidade do software.
- IEEE 1228. Plans de seguridade do software.

Probas

A **proba** exhaustiva do software é impracticable xa que non se poden probar todas as posibilidades do seu funcionamento, mesmo en programas pequenos e sinxelos. O obxectivo das probas é a detección de **defectos,** bugs ou erros do software, polo que unha proba é un éxito se descobre un defecto. Trátase dunha actividade a posteriori, de detección de problemas no software e non de prevención.

O proceso de proba comeza coa xeración dun **plan de probas** en base á documentación sobre o proxecto e á documentación sobre o software a probar. A partir do devandito plan, detállanse as probas específicas, execútanse os **casos de proba** e obtéñense os resultados.

Os **resultados da proba** poden indicar a **existencia de erros** e nese caso haberá que documentar e notificar os erros co obxectivo de que sexan arranxados e se volvan a executar as probas.

Non se deben de facer plans de proba pensando que non hai defectos; hai que asumir que sempre os hai. As probas deben de planificarse e deseñarse de forma sistemática para poder detectar o máximo número e variedade de defectos co mínimo consumo de tempo e esforzo. Son unha tarefa tanto ou máis creativa que o desenvolvemento de software e débense evitar as probas non documentadas ou aquelas deseñadas sen poñer coidado como por exemplo as que se realizan sobre a marcha.

As probas deben centrarse en dous obxectivos: probar se **o software non fai o que debe**, e probar se o software **fai o que non debe**, é dicir, se provoca efectos secundarios adversos; é habitual esquecer este último obxectivo.

Tipos de probas

Existen distintas maneiras de clasificar as probas non excluíntes entre si:

- **Funcionais ou non funcionais**. As primeiras están destinadas a comprobar algún requisito funcional do software e as segundas non. Exemplos de probas non funcionais: as que permiten indicar o esforzo requirido para aprender a manexar ese software, para analizar o código e localizar un fallo, para soportar cambios e facilitar as probas asociadas a eses cambios, ou para migrar o software a outro entorno.
- **Manuais e probas automáticas**. As primeiras realizaranse seguindo un plan detallado e estruturado pero sen dispoñer dun proceso automático para executalas. As segundas dispoñen dun proceso automático que pode repetirse cantas veces se queira de forma cómoda.
- **Dinámicas e probas estáticas**. As primeiras realízanse mentres o software se está executando e as segundas non.

Existen operacións que realizan certas ferramentas sobre o código para detectar defectos e que non son estritamente probas, como por exemplo:

- **Validación de código**. Permite garantir que o código cumpre algún estándar da linguaxe.
- **Depuración** (*Debugging*). Permite detectar o código que dá resultados erróneos na execución. Os contornos de desenvolvemento dispoñen de ferramentas automáticas de depuración que permiten executar o código de maneira controlada polo programador e permitindo que o programador realice a execución liña a liña ou defina puntos de interrupción para que a execución se pare nese punto; cando se para a execución, o pro-

gramador pode inspeccionar variables, expresións ou a pila e realizar modificacións sobre elas para ver como se comporta a execución.

- **Análise de liñas de código**. Permite detectar por exemplo: código repetido en varios sitios, revisar o código que está documentado e que facilitará a tarefa de xeración automática de documentación (Javadoc en Java), encontrar liñas de código comentado que ocupan espazo aínda que no se executen e que poden eliminarse utilizando un sistema de control de versións.

Probas unitarias

Son **probas funcionais** realizadas por persoal técnico que permiten detectar problemas nun módulo individual e elemental como por exemplo, un método dunha clase ou unha clase. Centran a súa actividade en exercitar a lóxica do módulo seguindo a estrutura do código (técnica de caixa branca) e as funcións que debe realizar o módulo atendendo ás entradas e saídas (técnica de caixa negra).

A **porcentaxe de código probado** mediante probas unitarias denomínase **cobertura do software**.

Os contornos de desenvolvemento dispoñen de ferramentas para deseñar e executar probas unitarias. Dentro das ferramentas dispoñibles no mercado destaca a familia XUnit: JUnit para Java, CppUnit para C++, PhpUnit para PHP e NUnit para .Net e Mono.

Probas de integración

Son **probas funcionais** realizadas por persoal técnico que permiten detectar problemas entre módulos relacionados e probados anteriormente de forma individual mediante probas unitarias. Deben ter en conta os mecanismos de ensamblaxe de módulos fixados na estrutura do programa, é dicir, as interfaces entre compoñentes da arquitectura do software. As probas de unidade e de integración solápanse e mestúranse no tempo normalmente.

Existen distintos tipos de probas de integración en función da orde de integración. A orde de integración elixida afecta a diversos factores, como: a forma de preparar casos, as ferramentas necesarias, a orde de codificar e probar os módulos, o custo da depuración, e o custo de preparación de casos. Os tipos fundamentais de integración son os seguintes:

- **Integración incremental**. Combínase o seguinte módulo que se debe probar co conxunto de módulos que xa foron probados. Existen dous tipos fundamentais:
    - **Ascendente**. Comézase polos módulos folla.
    - **Descendente**. Comézase polo módulo raíz.

- **Integración non incremental**. Próbase cada módulo por separado e logo intégranse todos dunha vez e próbase o programa completo. Denomínase tamén *Big-Bang* porque o número de módulos crece instantaneamente.

Integración incremental ascendente

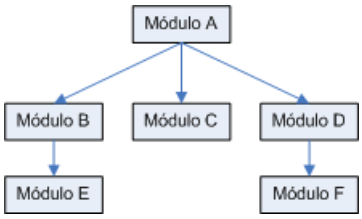
Na integración incremental ascendente empézase combinando os módulos de máis baixo nivel. As características da integración ascendente son:

- Pódese traballar con módulos de forma individual ou combinar os módulos de baixo nivel en grupos que realizan algunha función específica co obxectivo de reducir o número de pasos de integración.
- Escríbese para cada grupo un **módulo impulsor ou condutor** que é un módulo escrito para permitir simular a chamada aos módulos, introducir os datos de proba a través dos parámetros de entrada e recoller os resultados a través dos parámetros de saída.
- Próbase cada grupo empregando o seu impulsor.
- Elimínanse os módulos impulsores de cada grupo e substitúense polos módulos do nivel superior da xerarquía.

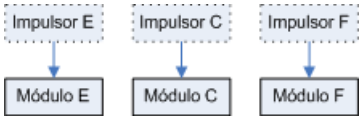
A construción de módulos impulsores non adoita ser moi complexa. Esta facilidade levou á creación de ferramentas automáticas capaces de realizar os labores dun impulsor. Normalmente están formados por:

- Instrucións para obter os datos necesarios para pasarlle ao módulo a probar.
- A chamada ao módulo a probar.
- Instrucións necesarias para mostrar os resultados devoltos polo módulo a probar.

Por exemplo, as etapas de integración serían 6 no caso de contar cos módulos seguintes.

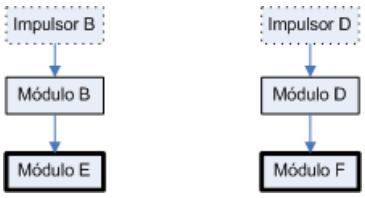


Etapas 1,2 e 3: Realízanse as probas unitarias de E, C e F. Os impulsores represéntanse cun borde punteado na seguinte gráfica.

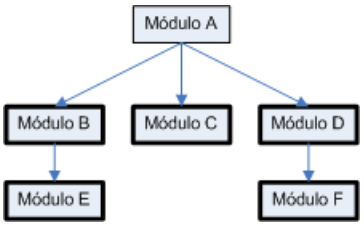


Etapas 4 e 5: Realízanse por unha parte e de forma simultánea, a proba de unidade de B e a de integración (ou de interface) B-E, e por outra e de forma simultánea, a proba da

unidade D e a interface D-F. Os módulos xa probados en etapas anteriores represéntanse cun borde groso na seguinte gráfica.



Etapa 6: Incorpórase o módulo A e próbase o programa completo, que non require impulsores, xa que todo o código de xestión da entrada e saída do programa está presente.



Integración incremental descendente

A integración incremental descendente comeza co módulo principal (o de maior nivel ou módulo raíz) e vai incorporando módulos subordinados progresivamente. Non existe unha regra xeral para determinar os módulos subordinados que convén incorporar primeiro. Como recomendación débense incorporar canto antes as partes críticas e os módulos de Entrada/Saída. Existen dúas ordes fundamentais de integración descendente:

- **Primeiro en profundidade**: vanse completando ramas da árbore modular. No exemplo anterior, a secuencia de módulos sería A-B-E-C-D-F.
- **Primeiro en anchura**: vanse completando niveis horizontais de xerarquía modular. No exemplo anterior, a secuencia de módulos sería A-B-C-D-E-F.

As características da integración descendente son:

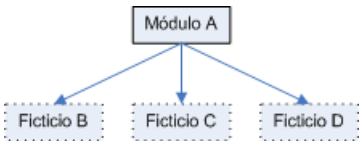
- módulo raíz é o primeiro en probarse e escríbense **módulos ficticios** (*stubs*), para simular a presenza dos subordinados ausentes, que serán chamados polo módulo raíz.
- Unha vez probado o módulo raíz substitúese un dos subordinados ficticios polo módulo correspondente segundo a orde elixida, e incorpóranse ficticios para recoller as chamadas do último incorporado.
- Repítese o proceso detallado para o módulo raíz, é dicir, execútanse as correspondentes probas cada vez que se incorpora un módulo novo e, ao terminar cada proba, substitúese un ficticio polo seu correspondente real. Convén repetir algúns casos de proba de execucións anteriores para asegurarse de que non se introduciu ningún defecto novo.

A codificación de módulos ficticios subordinados é máis complicada que a creación de impulsores. Os ficticios deben simular que recollen o control da chamada e que fan algo cos parámetros que se lles pasan. Existen diversos niveis de sofisticación dos ficticios:

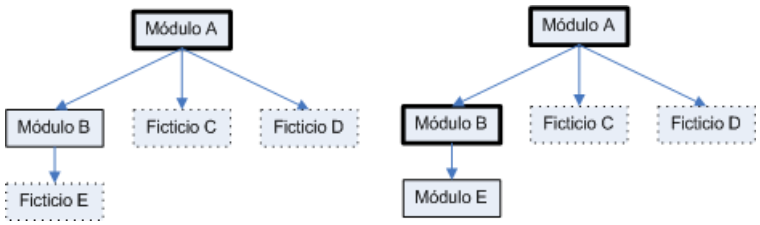
- Módulos que só mostran unha mensaxe de traza. Por exemplo, printf("Executouse o módulo 1");
- Módulos que mostran os parámetros que se lles pasan. Por exemplo, recibe o parámetro x e o mostra con printf("%d", x);
- Módulos que devolven un valor que non depende dos parámetros que se pasen como entrada (sempre devolve o mesmo valor, ou un valor aleatorio, etc.). Por exemplo, return(5); independentemente dos parámetros que reciba.
- Módulos que, en función dos parámetros pasados, devolven un valor de saída que máis ou menos corresponda a dita entrada. Por exemplo, devolver un valor utilizando as entradas para buscar nun array bidimensional, return(Táboa[x][y]); cando recibe x e y.

Por exemplo, utilizando o mesmo deseño de módulos que para a integración ascendente, e utilizando a orde primeiro en profundidade, serían necesarias 6 etapas.

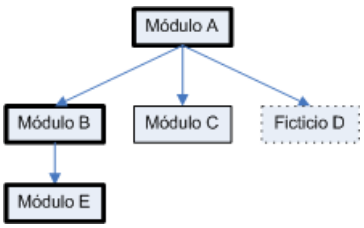
Etapa 1: Realízase a proba unitaria de A. Os ficticios represéntanse cun borde punteado na seguinte gráfica.



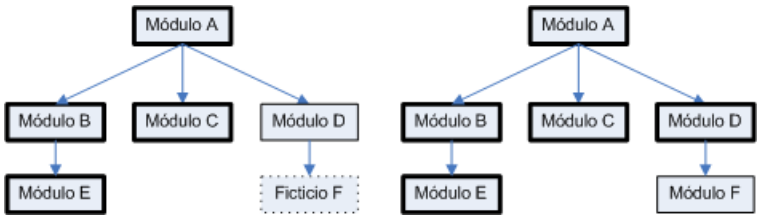
Etapas 2 e 3: Realízanse simultaneamente a proba de unidade de B e a de integración (ou de interface) A-B, e a continuación realízase simultaneamente a proba da unidade E e a interface B-E. Os módulos probados en etapas anteriores represéntanse cun borde groso na seguinte gráfica.



Etapa 4: Realízase simultaneamente a proba do módulo C e a interface A-C.



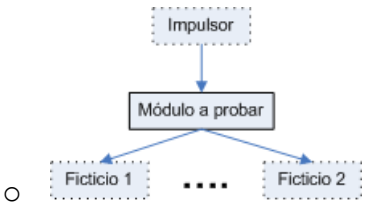
Etapas 5 e 6: Realízase simultaneamente a proba do módulo D e a interface A-D e a continuación a proba do módulo F e a interface D-F, quedando toda a integración probada.



Integración non incremental

A integración non incremental é o único caso no que as probas de unidade e integración están totalmente separadas. As características da integración non incremental son:

- Cada módulo require para ser probado:
  - Dun **módulo impulsor** que transmite os datos de proba ao módulo e mostra os resultados dos devanditos casos de proba.
  - Os **módulos ficticios** necesarios para simular a función de cada módulo subordinado ao módulo que imos probar.



- Despois de probar cada módulo por separado, ensámblanse todos eles dunha soa vez para formar o programa completo.

Comparación entre os distintos tipos de integración

A comparación entre as vantaxes da integración non incremental e a incremental queda reflectida na seguinte táboa:

Vantaxes da integración non incremental	Vantaxes da integración incremental
▪ Require menos tempo de máquina para as probas, xa que se proba dunha soa vez a combinación dos módulos. ▪ Existen máis oportunidades de probar módulos en paralelo.	▪ Require menos traballo, xa que se escriben menos módulos impulsores e ficticios. ▪ Os defectos e erros nas interfaces detéctanse antes, xa que se empeza antes a probar as unións entre os módulos. ▪ A depuración é moito máis fácil, xa que, de detectar os síntomas dun defecto nun paso da integración, hai que atribuílo moi proba-

	blemente ao último módulo incorporado. ▪ Examínase con maior detalle o programa, ao ir comprobando cada interface aos poucos.

A comparación entre as características da integración incremental ascendente e descendente queda reflectida na seguinte táboa:

Integración incremental ascendente	Integración incremental descendente
▪ É vantaxosa cando hai defectos en niveis inferiores do programa. ▪ As entradas son máis fáciles de crear. ▪ O programa, como entidade, non existe até incorporar o último módulo. ▪ Requírense módulos impulsores. Os módulos impulsores son fáciles de crear. ▪ É máis fácil observar os resultados das probas.	▪ É vantaxosa cando hai fallos nos niveis superiores do programa. ▪ Antes de incorporar E/S, é difícil representar casos e as entradas poden ser difíciles de crear. Tras incorporar funcións de E/S, é fácil a representación de casos. ▪ Antes de incorporar o último módulo, vese a estrutura previa do programa. ▪ Require ficticios subordinados. Os ficticios subordinados non son fáciles de crear. ▪ Difícil observar resultados. ▪ Induce a atrasar a terminación da proba dalgúns módulos.

A selección dunha estratexia de integración depende das características do software e, ás veces, da planificación do proxecto. Algunhas recomendacións poderían ser:

- Identificar e probar canto antes aqueles módulos considerados críticos ou problemáticos.
- En xeral, o mellor compromiso pode ser un enfoque combinado (ás veces denominado *sandwich*) que consiste en:
    - Usar a integración descendente para os niveis superiores da estrutura do programa.
    - Utilízase simultaneamente a ascendente para os niveis subordinados.
    - Continúase ata que ambas as aproximacións se atopen nalgún nivel intermedio.

Probas de sistema ou implantación

Son probas realizadas por persoal técnico que permiten comprobar que o sistema completo, cumpre os requisitos funcionais e técnicos especificados e que se relaciona correctamente con outros sistemas.

Ademais realízanse outras probas como:

- **Probas de sobrecarga ou estrés**: Permiten avaliar o comportamento do sistema ao sometelo a unha situación límite como, por exemplo, demanda excesiva de peticións, utilización da máxima cantidade de memoria, traballar con pouca memoria, ter moitos usuarios realizando ao mesmo tempo a mesma operación, utilizar o máximo volume de datos.
- **Probas de compatibilidade**: Permiten probar o sistema en diferentes contornos, medios ou sistemas operativos e ver se hai fallos no aspecto ou no funcionamento.

- **Probas de seguridade e acceso a datos**: Permiten probar como responde o sistema ante ataques externos como por exemplo irrupcións non autorizadas ou camuflaxe de código malicioso nunha entrada de datos.
- **Probas de recuperación e tolerancia a fallos**: Permiten provocar anomalías externas como fallos eléctricos, de dispositivos, de software externo ou de comunicacións, e ver que o sistema se recupera sen perda de datos e sen fallos de integridade e o tempo que lle leva facelo.

Probas de validación ou aceptación

Son probas realizadas por persoal non técnico e serven para comprobar se o produto final se axusta aos requisitos iniciais do software e están baseadas nas accións visibles para o usuario e nas saídas do software que este pode recoñecer.

Poden existir probas alfa e/ou probas beta:

- As **probas alfa** asócianse normalmente ao software contratado. Realízaas o cliente nun contorno controlado baixo a supervisión da empresa que desenvolve o software. O desenvolvedor de software tomará nota dos posibles erros e problemas de uso.
- As **probas beta** asócianse normalmente ao software de interese xeral como por exemplo os paquetes ofimáticos pero tamén poden utilizarse para software contratado. Realízanas usuarios de confianza no seu contorno real de traballo sen control directo da empresa de software. O usuario informará dos resultados das probas.

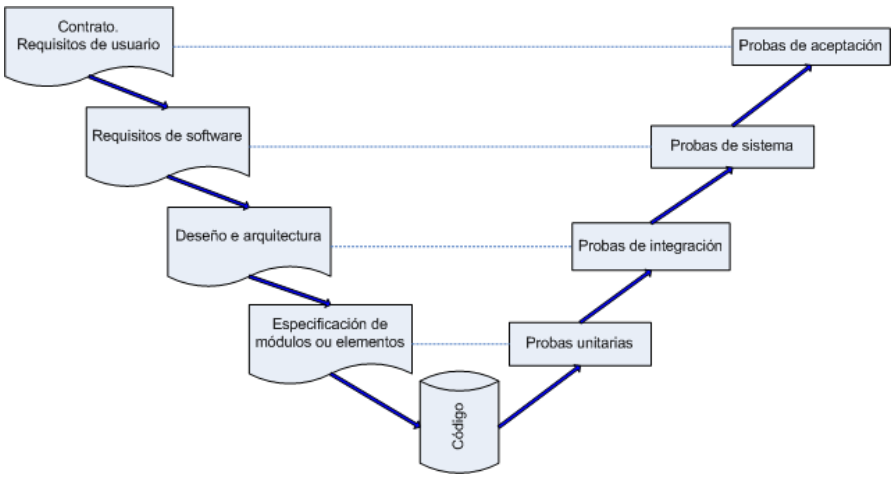
Probas de regresión

Son probas realizadas por persoal técnico para evitar efectos colaterais. Aplícanse cada vez que se fai un cambio no software para verificar que non aparecen comportamentos non desexados ou erros noutros módulos ou partes do software.

Un caso particular de cambio no software dáse cando se agrega un novo módulo nas probas de integración incremental. Pode ocorrer que un módulo que funcionaba ben deixe de facelo pola incorporación doutro. Neste caso as probas de regresión consistirán en volver a aplicar un subconxunto significativo das probas realizadas aos módulos xa integrados.

Estratexia de aplicación das probas no ciclo de vida clásico

A estratexia de aplicación e a planificación das probas pretende integrar o deseño dos casos de proba nunha serie de pasos ben coordinados, a través da creación de distintos niveis de proba, con diferentes obxectivos. En xeral a estratexia de probas adoita seguir durante o ciclo de vida do software as etapas que se mostran na seguinte imaxe.

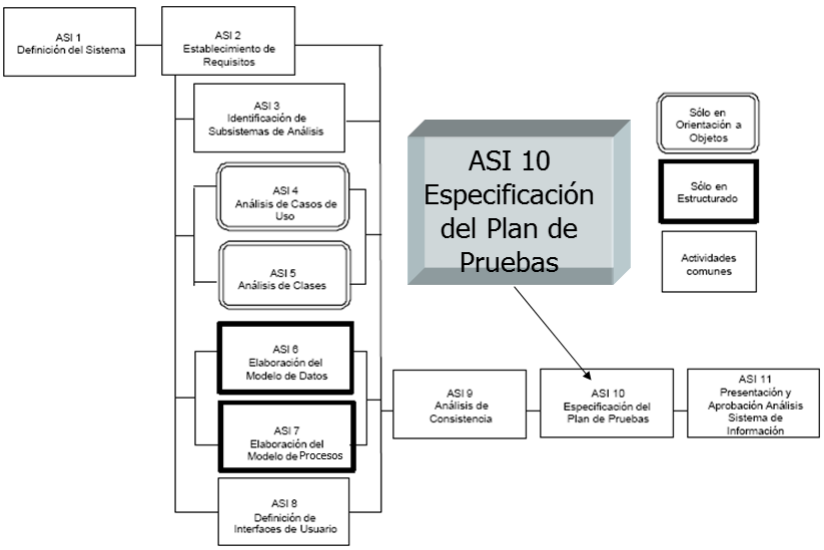


Estratexia de aplicación das probas en Métrica V. 3

A metodoloxía MÉTRICA Versión 3 ofrece ás organizacións un instrumento útil para a sistematización das actividades que dan soporte ao ciclo de vida do software. Pode ser utilizada libremente coa única restrición de citar a fonte da súa propiedade intelectual, é dicir, o Ministerio de Facenda e Administracións Públicas. En http://administracionelectronica.gob.es/pae_Home/pae_Documentacion/pae_Metodolog/pae_Metrica_v3.html#.VG8NkOJ3Ofs poden verse os documentos que compoñen a metodoloxía MÉTRICA V. 3. Nas seguintes imaxes extractadas dos documentos pdf que se poden baixar da páxina web oficial, móstrase a relación de actividades dos diferentes procesos do sistema de información, tanto para desenvolvementos estruturados como para desenvolvementos orientados a obxectos, distinguindo as que se poden realizar en paralelo de aquelas que se deben de realizar de forma secuencial.

As probas na análise (ASI)

O plan de proba é un produto formal que define os obxectivos da proba de todo o sistema de información, establece e coordina unha estratexia de traballo e prové do marco adecuado para elaborar unha planificación paso a paso das actividades da proba. Este plan vaise completando e detallando a medida que se avanza nos restantes procesos do ciclo de vida do software.

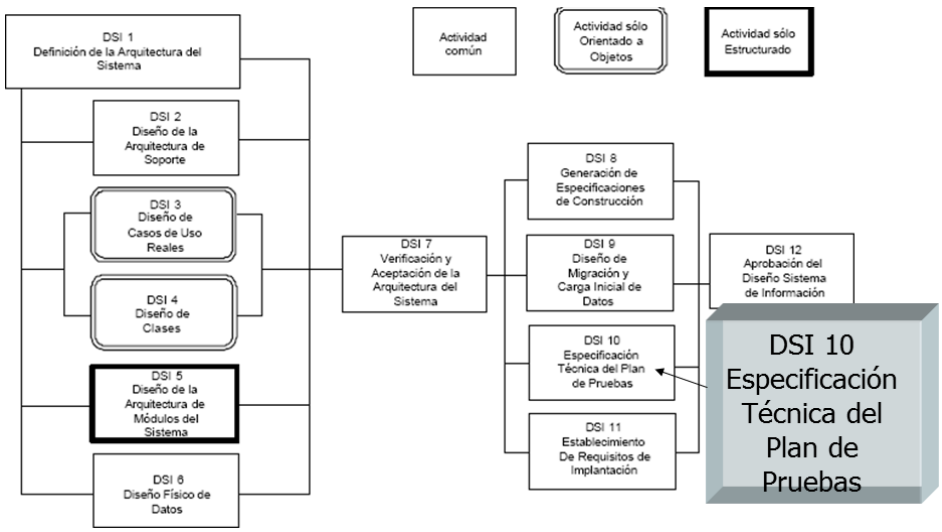


ASI 10: Especificación del Plan de Pruebas

Tarea		Productos	Técnicas y Prácticas	Participantes
ASI 10.1	Definición del Alcance de las Pruebas	- Plan de Pruebas	- Sesiones de Trabajo	- Jefe de Proyecto - Analistas - Equipo de Soporte Técnico - Usuarios Expertos
ASI 10.2	Definición de Requisitos del Entorno de Pruebas	- Plan de Pruebas	- Sesiones de Trabajo	- Jefe de Proyecto - Analistas - Equipo de Soporte Técnico - Usuarios Expertos
ASI 10.3	Definición de las Pruebas de Aceptación del Sistema	- Plan de Pruebas	- Sesiones de Trabajo	- Jefe de Proyecto - Analistas - Equipo de Soporte Técnico - Usuarios Expertos

As probas no deseño (DSI)

Faise a especificación técnica do plan de probas, é dicir, detállase o plan de probas do sistema de información para cada un dos niveis de probas establecido: probas unitarias, integración, sistema, implantación e aceptación.

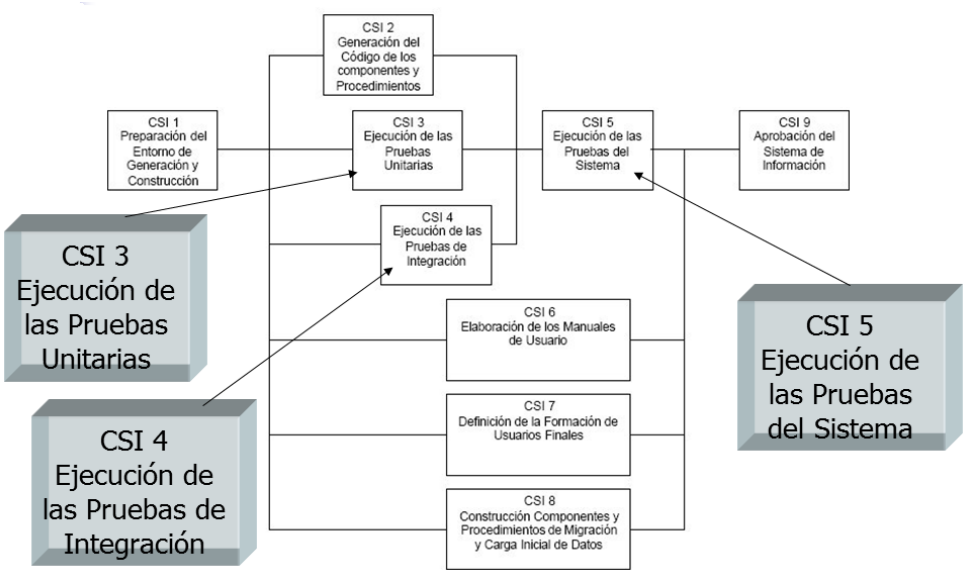


DSI 10: Especificación Técnica del Plan de Pruebas

Tarea		Productos	Técnicas y Prácticas	Participantes
DSI 10.1	Especificación del Entorno de Pruebas	- Plan de Pruebas: ○ Especificación del Entorno de Pruebas		- Equipo de Arquitectura - Equipo de Soporte Técnico - Equipo del Proyecto - Equipo de Seguridad
DSI 10.2	Especificación Técnica de Niveles de Prueba	- Plan de Pruebas: ○ Especificación Técnica de Niveles de Prueba		- Jefe de Proyecto - Analistas - Usuarios Expertos
DSI 10.3	Revisión de la Planificación de Pruebas	- Plan de Pruebas: ○ Planificación de las Pruebas		- Jefe de Proyecto

As probas na construción (CSI)

Execútanse as probas deseñadas na etapa anterior.



Realízanse as probas unitarias de cada un dos compoñentes do sistema despois de codificarse co obxectivo de comprobar que a estrutura é correcta e se axustan á funcionalidade establecida.

CSI 3: Ejecución de las Pruebas Unitarias

Tarea		Productos	Técnicas y Prácticas	Participantes
CSI 3.1	Preparación del Entorno de Pruebas Unitarias	- Entorno de pruebas unitarias		- Técnicos de Sistemas - Programadores
CSI 3.2	Realización y evaluación de las Pruebas Unitarias	- Resultado de las pruebas unitarias	- Pruebas Unitarias	- Programadores

Realízanse as probas de integración para verificar se os compoñentes ou subsistemas interactúan correctamente a través das súas interfaces, tanto internas como externas, cobren a funcionalidade establecida e se axustan aos requisitos especificados.

CSI 4: Ejecución de las Pruebas de Integración

Tarea		Productos	Técnicas y Prácticas	Participantes
CSI 4.1	Preparación del Entorno de las Pruebas de Integración	- Entorno de Pruebas de Integración		- Técnicos de Sistemas - Técnicos de Comunicaciones - Equipo de Arquitectura - Equipo del Proyecto
CSI 4.2	Realización de las Pruebas de Integración	- Resultado de las Pruebas de Integración	- Pruebas de integración	- Equipo del Proyecto
CSI 4.3	Evaluación del Resultado de las Pruebas de Integración	- Evaluación del Resultado de las Pruebas de Integración		- Analistas

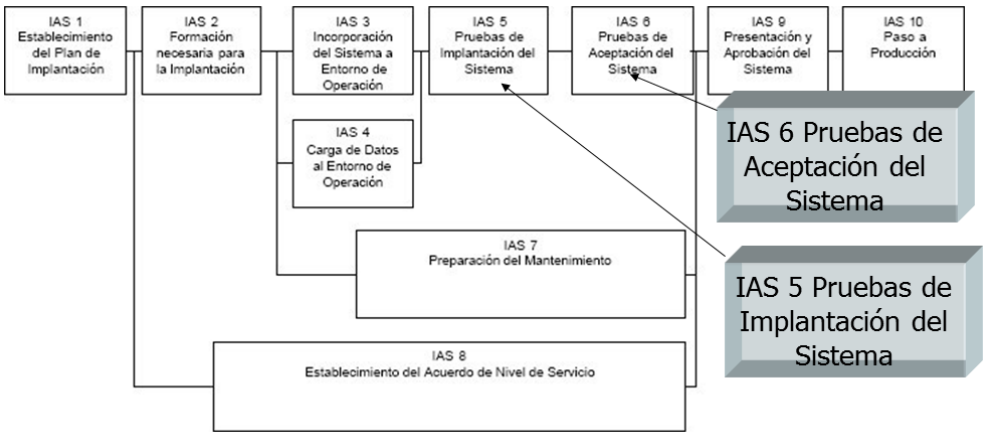
Realízanse as probas do sistema para comprobar a integración do sistema de información completo, verificando o funcionamento correcto das interfaces entre os distintos subsistemas que o compoñen e co resto de sistemas de información cos que se comunica.

CSI 5: Ejecución de las Pruebas del Sistema

Tarea		Productos	Técnicas y Prácticas	Participantes
CSI 5.1	Preparación del Entorno de las Pruebas del Sistema	- Entorno de Pruebas del Sistema		- Técnicos de Sistemas - Técnicos de Comunicaciones - Equipo de Arquitectura - Equipo del Proyecto
CSI 5.2	Realización de las Pruebas del Sistema	- Resultado de las Pruebas del Sistema	- Pruebas del Sistema	- Equipo del Proyecto
CSI 5.3	Evaluación del Resultado de las Pruebas del Sistema	- Evaluación del Resultado de las Pruebas del Sistema		- Analistas - Jefe de Proyecto

As probas na implantación (IAS)

Realízanse as probas de implantación do sistema para comprobar o funcionamento do mesmo no entorno de operación e realízanse as probas de aceptación do sistema co fin de permitir que o usuario determine que acepta o sistema.



IAS 5: Pruebas de Implantación del Sistema

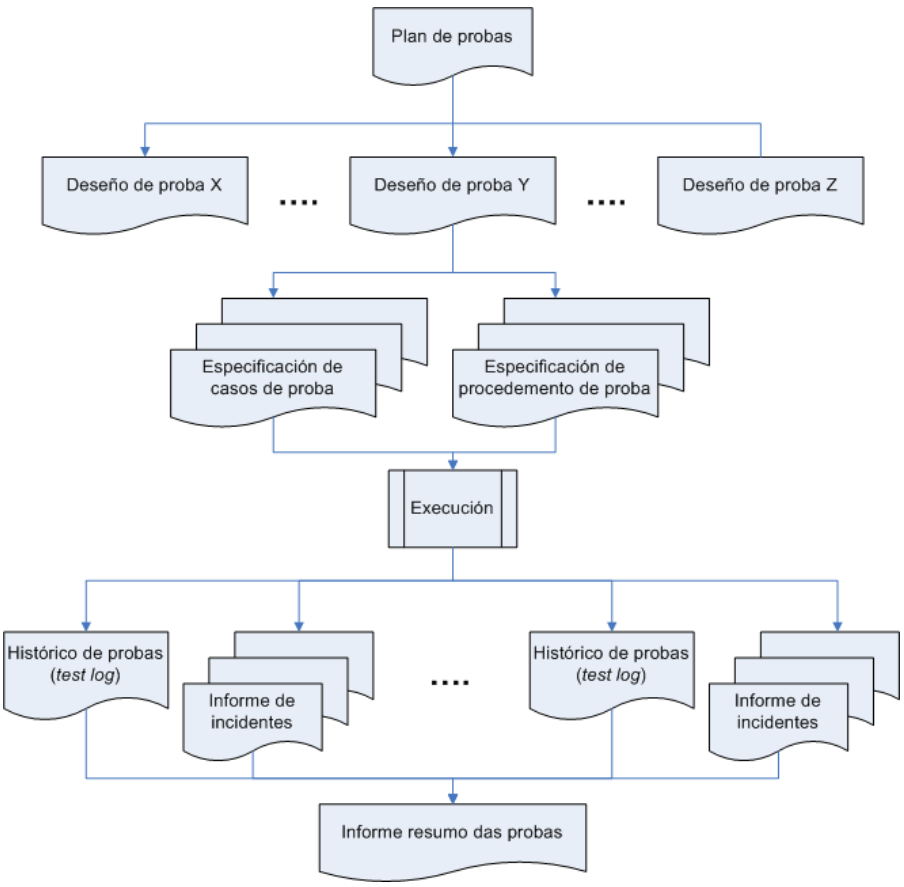
Tarea		Productos	Técnicas y Prácticas	Participantes
IAS 5.1	Preparación de las Pruebas de Implantación	- Plan de pruebas		- Jefe de Proyecto - Responsable de Implantación
IAS 5.2	Realización de las Pruebas de Implantación	- Resultado de las pruebas de implantación	- Pruebas de implantación	- Equipo de Implantación
IAS 5.3	Evaluación del resultado de las Pruebas de Implantación	- Evaluación del resultado de las pruebas de implantación		- Jefe de Proyecto - Responsable de Implantación

IAS 6: Pruebas de Aceptación del Sistema

Tarea		Productos	Técnicas y Prácticas	Participantes
IAS 6.1	Preparación de las Pruebas de Aceptación	- Plan de Pruebas		- Jefe de Proyecto - Directores de los Usuarios
IAS 6.2	Realización de las Pruebas de Aceptación	- Resultado de las Pruebas de Aceptación	- Pruebas de Aceptación	- Usuarios Expertos
IAS 6.3	Evaluación del resultado de las Pruebas de Aceptación	- Evaluación del Resultado de las Pruebas de Aceptación		- Jefe de Proyecto - Directores de los Usuarios

Documentación do deseño das probas

A documentación das probas é necesaria para unha boa organización das mesmas, así como para asegurar a súa reutilización. Segundo o estándar IEEE 829, os documentos relacionados coas probas asócianse ás distintas fases das probas segundo se indica na gráfica seguinte.



O primeiro paso sitúase na planificación xeral do esforzo de proba (**plan de probas**) que inclúe o alcance do plan, os recursos a utilizar, a planificación das probas e as actividades a realizar. O segundo paso consiste no **deseño das probas** que xorde da ampliación e detalle do plan de probas e no que se especifican as características das diferentes probas. A partir deste deseño, especifícanse cada un dos casos de proba mencionados brevemente no deseño da proba e especifícase como proceder en detalle e desenvolver a execución dos devanditos casos (procedementos de proba). Tanto as especificacións de casos de proba como as especificacións dos procedementos deben ser os documentos básicos para a **execución das probas**. O último paso é o informe co **resumo das probas** no que se reflicten os resultados das actividades de proba e se achega unha avaliación do software baseada nos devanditos resultados.

A documentación da execución das probas é fundamental para a eficacia na detección e corrección de defectos, así como para deixar constancia dos resultados da execución das probas. Os documentos xerados para cada execución son:

- **Histórico de probas**: Documenta os feitos relevantes ocorridos durante a execución das probas.
- **Informe de incidentes**: Documenta cada incidente ocorrido na proba e que requira unha posterior investigación.