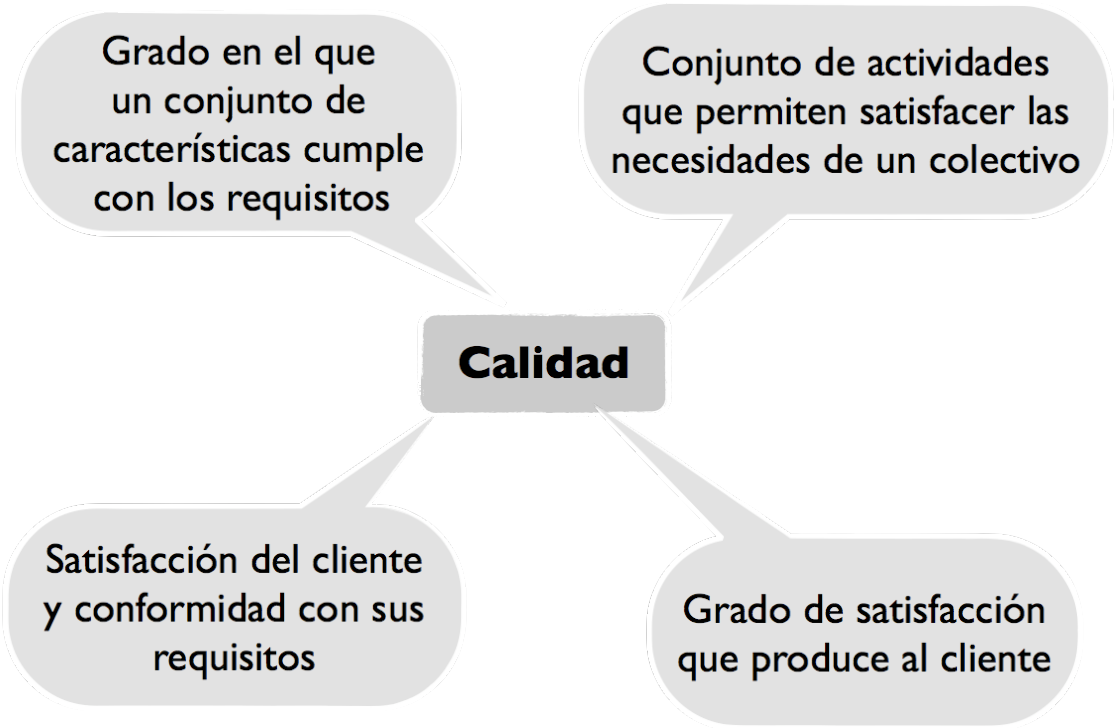
Métricas de Calidad de Software

Integrantes:
- Betzabeth Pereira
- Farid Ayaach
- Henry Quintero
- Ismael Granadillo
- Jomar Bustamante

Definiciones



Definiciones

- *Medida*
  Proporciona una indicación cuantitativa de la cantidad, dimensiones o tamaño de algunos atributos de un producto.

- *Medición*
  Acto de determinar una medida.

- *Métrica*
  Es una medida del grado en que un sistema, componente o proceso posee un atributo dado.

Métricas de Software

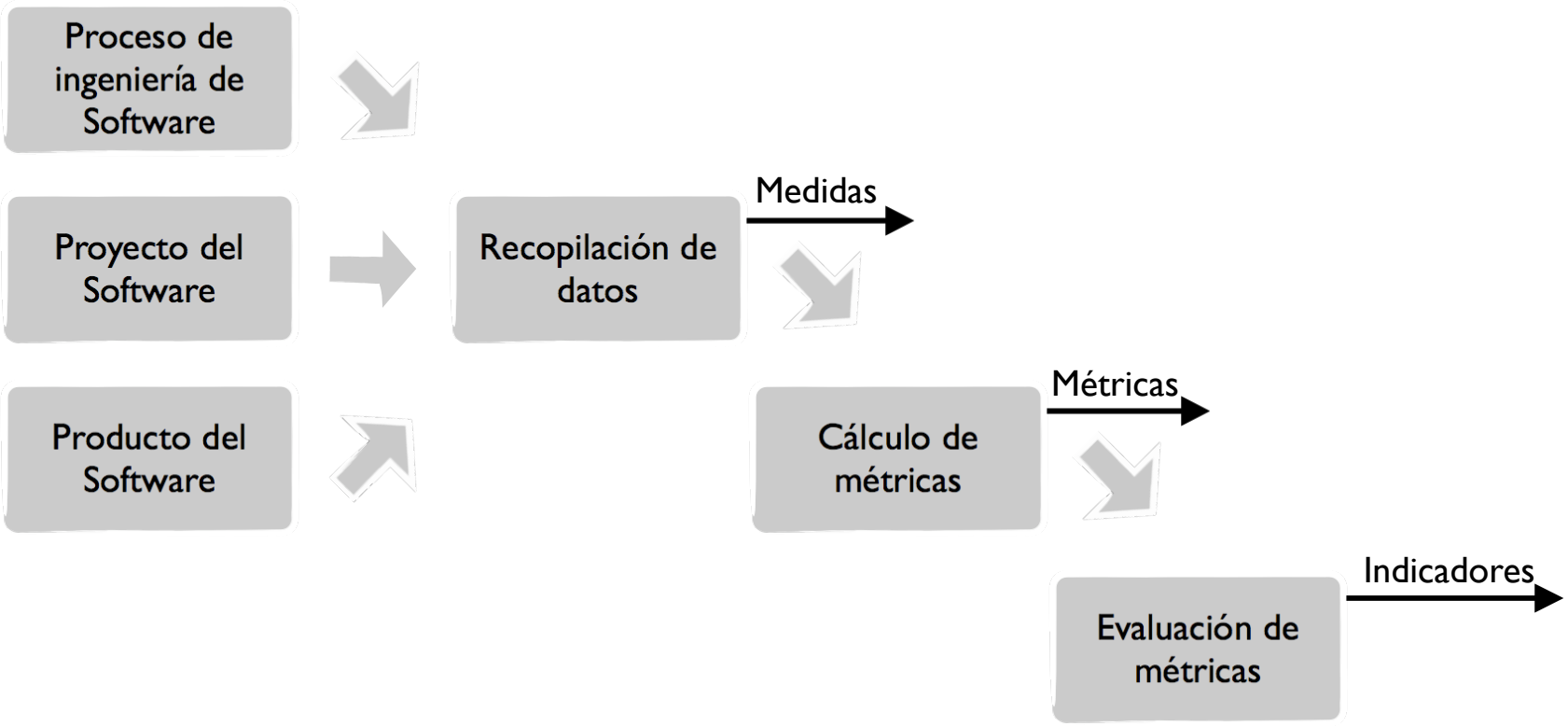
Aplicación continua de mediciones en el proceso de desarrollo del software y sus productos, para suministrar información relevante a tiempo

mejorar tanto el procesos como los productos

- Las métricas del Software comprenden un amplio rango de actividades diversas, estas son algunas:
- Aseguramiento y control de calidad
- Modelos de fiabilidad
- Modelos y evaluación de ejecución
- Modelos y medidas de productividad

Métricas de Software

Proceso de recopilación de métricas de Software



Clasificación de las métricas de Software

Según los criterios:

de complejidad	Métricas que definen la medición de la complejidad: volumen, tamaño, anidaciones, y configuración.
de calidad	Métricas que definen la calidad del software: exactitud, estructuración o modularidad, pruebas, mantenimiento.
de competencia	Métricas que intentan valorar o medir las actividades de productividad de los programadores con respecto a su certeza, rapidez, eficiencia y competencia
de desempeño	Métricas que miden la conducta de módulos y sistemas de un software, bajo la supervisión del SO o hardware.
estilizadas	Métricas de experimentación y de preferencia: estilo de código, convenciones, limitaciones, etc.

Clasificación de las métricas de Software

Según el contexto en que se aplican:

- *Métricas de proceso*
  - Se recopilan de todos los proyectos, y durante un largo periodo de tiempo
  - Caracterizados por:
    - Control y ejecución del proyecto.
    - Medición de tiempos de las fases.
- *Métricas de proyecto*
  - Permiten evaluar el estado del proyecto.
  - Permiten seguir la pista de los riesgos.
- *Métricas de producto*
  - Se centran en las características del software y no en como fue producido.
  - También son productos los artefactos, documentos, modelos, y componentes que conforman el software.
  - Se miden cosas como el tamaño, la calidad, la totalidad, la volatilidad, y el esfuerzo.

Métricas de Calidad

- Principal objetivo de los ingenieros de software es producir sistemas, aplicaciones o productos de alta calidad.

- Para las evaluaciones que se quieran obtener es necesario la utilización de medidas técnicas, que evalúan la calidad de manera objetiva.

Métricas de Calidad - Modelos conocidos

Modelo de MCCALL (1977)

- Describe la calidad como un concepto elaborado mediante relaciones jerárquicas entre factores de calidad, en base a criterios
- Los factores de calidad se concentran en tres aspectos importantes de un producto de software: características operativas, capacidad de cambios y adaptabilidad a nuevos entornos.
- Identifica una serie de criterios, tales como rastreabilidad, simplicidad, capacidad de expansión, etc.
- Las métricas desarrolladas están relacionadas con los factores de calidad y la relación que se establece se mide en función del grado de cumplimiento de los criterios.

Métricas de Calidad - Modelos conocidos

Modelo de MCCALL (1977)

Factor	Criterio
Correctitud	Rastreabilidad Completitud Consistencia
Confiabilidad	Consistencia Exactitud Tolerancia a fallas
Eficiencia	Eficiencia de ejecución Eficiencia de almacenamiento
Integridad	Control de acceso Auditoría de acceso
Usabilidad	Operabilidad Entrenamiento Comunicación
Interoperabilidad	Modularidad Similitud de comunicación Similitud de datos.

Criterios asociados a los factores de calidad

Factor	Criterio
Mantenibilidad	Simplicidad Concreción
Capacidad de Prueba	Simplicidad Instrumentación Auto-descriptividad Modularidad
Flexibilidad	Auto-descriptividad Capacidad de expansión Generalidad Modularidad
Portabilidad	Auto-descriptividad Independencia del sistema Independencia de máquina
Reusabilidad	Auto-descriptividad Generalidad Modularidad Independencia del sistema Independencia de máquina

Métricas de Calidad - Modelos conocidos

Modelo de FURPS (1987)

- Modelo desarrollado por Hewlett-Packard (HP) en 1987, desarrollando un conjunto de factores de calidad de software y sus respectivos atributos.

- Funcionalidad *(Functionality), usabilidad (Usability), confiabilidad (Reliability), desempeño (Performance) y capacidad de soporte (Supportability).*

- *Basado en el modelo de MCCALL.*

- Se utilizan para establecer métricas de la calidad para todas las actividades del proceso de desarrollo de un software, inclusive de un sistema de información.

Métricas de Calidad - Modelos conocidos

Modelo de FURPS (1987)

Factor	Criterio
Funcionalidad	Características y capacidades del programa Generalidad de las funciones Seguridad del sistema
Facilidad de Uso	Factores humanos Factores estéticos Consistencia de la interfaz Documentación
Confiabilidad	Frecuencia y severidad de las fallas Exactitud de las salidas Tiempo medio de fallos Capacidad de recuperación ante fallas Capacidad de predicción

Factor	Criterio
Rendimiento	Velocidad del procesamiento Tiempo de respuesta Consumo de recursos Rendimiento efectivo total Eficacia
Capacidad de Soporte	Extensibilidad Adaptabilidad Capacidad de pruebas Capacidad de configuración Compatibilidad Requisitos de instalación

Criterios asociados a los factores de calidad

Métricas de Calidad - Modelos conocidos

Modelo de DROMEY (1996)

- Resalta el hecho de que la calidad del producto es altamente determinada por los componentes del mismo *(incluyendo documentos de requerimientos, guías de usuarios, diseños, y código),*

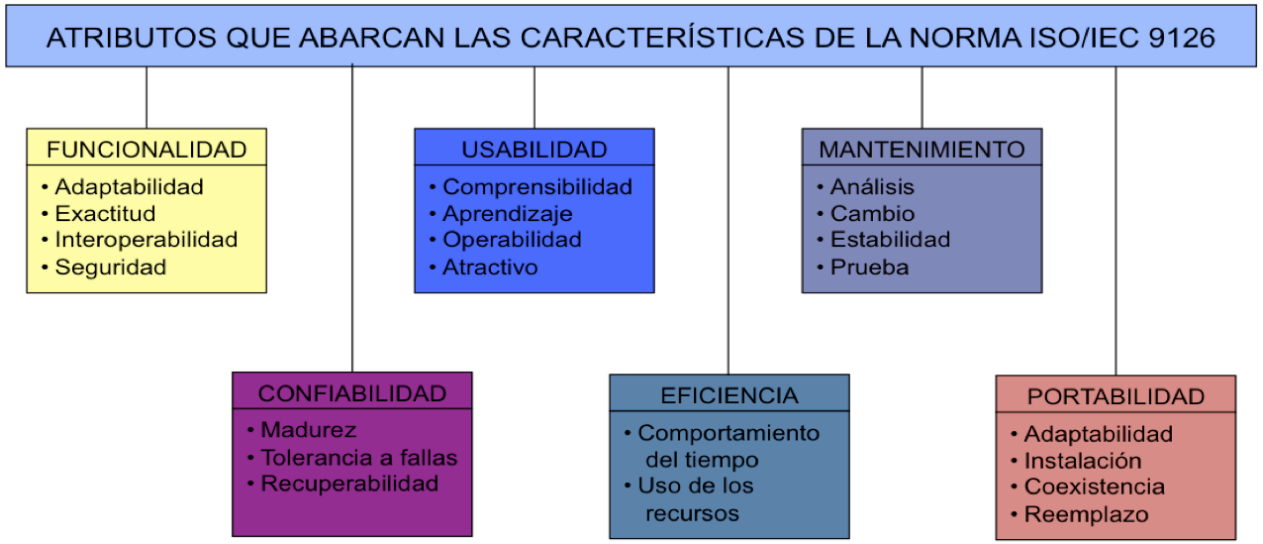
- Sugiere el uso de cuatro categorías que implican propiedades de calidad, que son: correctitud, internas, contextuales y descriptivas.

Factor	Criterio
Correctitud	Funcionalidad Confiabilidad
Internas	Mantenibilidad Eficiencia Confiabilidad
Contextuales	Mantenibilidad Reusabilidad Portabilidad Confiabilidad
Descriptivas	Mantenibilidad Reusabilidad Portabilidad Usabilidad

Criterios asociados a los factores de calidad

Métricas de Calidad - Modelos conocidos

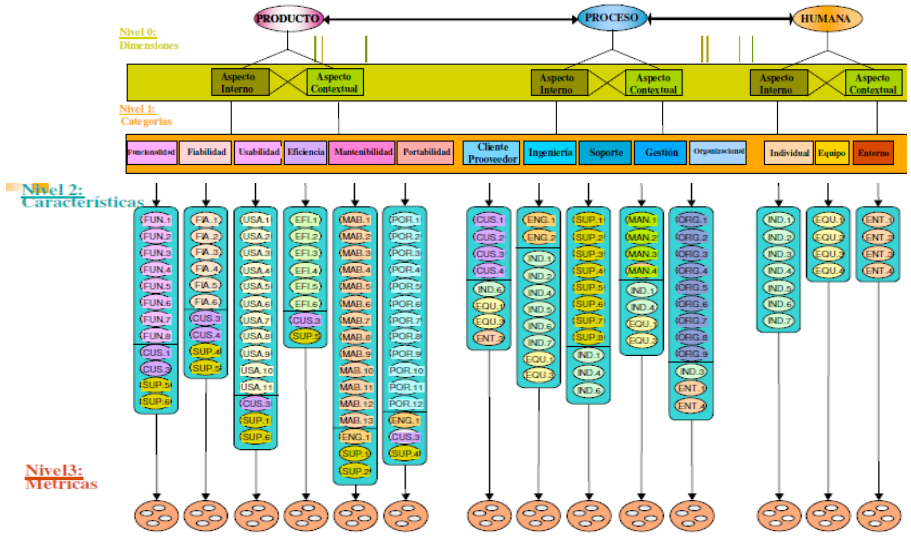
Normas ISO 9000
ISO/IEC 9126



Métricas de Calidad - Modelos conocidos

MOSCA (Modelo Sistémico de Calidad)

- Consta de 4 niveles: dimensiones, categorías, características y las métricas. En base de tres ramas: el producto, el proceso y la humana. Contiene un total de 715 métricas.



Métricas de Calidad - Modelos conocidos

MOSCA (Modelo Sistémico de Calidad)

Ejemplo de agrupación de métricas

Categoría	Características	
	Aspectos Contextuales del Producto	**Aspectos Internos del Producto**
Funcionalidad (FUN) Total de métricas: 46	FUN 1. Ajuste a los propósitos (16) FUN 2. Precisión (10) FUN 3. Interoperabilidad (7) FUN 4. Seguridad (2)	FUN 5. Correctitud (8) FUN 6. Estructurado (1) FUN 7. Encapsulado (1) FUN 8. Especificado (1)
	Sub-total de métricas: 35	Sub-total de métricas: 11
Fiabilidad (FIA) Total de métricas: 32	FIA 1. Madurez (17) FIA 2. Tolerancia a fallas (1) FIA 3. Recuperación (4)	FIA 4. Correctitud (8) FIA 5. Estructurado (1) FIA 6. Encapsulado (1)
	Sub-total de métricas: 22	Sub-total de métricas: 10
Usabilidad (USA) Total de métricas: 38	USA 1. Facilidad de comprensión (5) USA 2. Capacidad de Aprendizaje (9) USA 3. Interfaz Gráfica (5) USA 4. Operabilidad (13) USA 5. Conformidad con los estándares	USA 6. Completo (1) USA 7. Consistente (1) USA 8. Efectivo (1) USA 9. Especificado (1) USA 10. Documentado (1) USA 11. Auto-descriptivo (1)
	Sub-total de métricas: 32	Sub-total de métricas: 6

Métricas de Calidad - Modelos conocidos

MOSCA (Modelo Sistémico de Calidad)

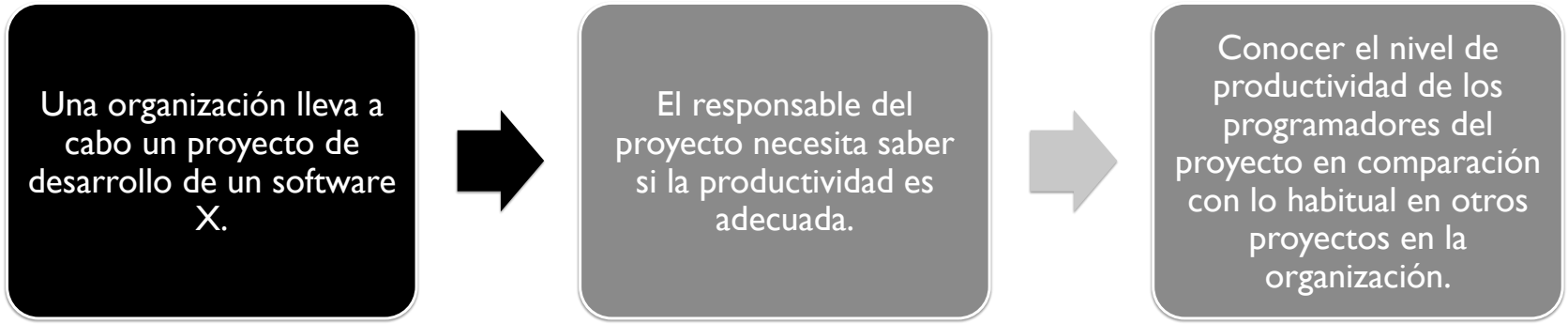
Ejemplo de métricas

Métrica	Número de componentes de acceso a base de datos.
Rangos para la métrica	≥ 8 → 5 5 – 7 → 4 3 - 4 → 3 1 - 2 → 2 0 → 1

Métrica	Generación de documentación en concordancia con los estándares y políticas establecidos.
Rangos para la métrica	SI ☐ NO ☐ N/A ☐ N/S ☐

Métricas de Calidad - Modelos conocidos

Ejemplo



Métricas de Calidad - Modelos conocidos

Ejemplo

Las métricas a utilizar podrían ser:

Directas	Indirectas	Indicadores
• LCF: líneas de código fuente escritas. • HPD: horas-programador diarias. • CHP: coste por hora-programador, en unidades monetarias.	• HPT: horas-programador totales. • LCFH: líneas de código fuente por hora de programador. • CTP: coste total actual del proyecto, en unidades monetarias. • CLCF: coste por línea de código fuente.	• PROD: productividad de los programadores.

Métricas de Calidad - Modelos conocidos

Ejemplo

La forma de obtenerlas viene dada por:

Directas	Indirectas	Indicadores
• LCF = Contar las líneas de código. • HPD = Contar cada día las horas dedicadas por los programadores al proyecto. • CHP = Consultar el plan de proyecto.	• HPT = Σ HPD • LCFH = LCF/HPT • CTP = CHP*HPT • CLCF = LCF/CTP	• PROD: Establecer criterios o rangos de valores.

Software Libre y Calidad

- La calidad se ha convertido en uno de los elementos diferenciadores en el ámbito mundial entre las compañías desarrolladoras de sistemas de software. La búsqueda de la calidad de los sistemas ha propiciado la creación de modelos, *frameworks* y metodologías para evaluar y asegurar su calidad.

- El Software Libre también ha tenido un impulso que ha despertado un interés particular en sus herramientas y modelos de negocios, pero sobre todo en sus procesos de desarrollo.

- Pero, ¿cómo se relacionan estos dos conceptos (calidad y Software Libre)?

Software Libre y Calidad

- Nace entonces la necesidad de estimar la calidad de este tipo de herramientas. En el 2006 surge el Software Quality Observatory for Open Source Software (SQO-OSS).

- SQO-OSS desarrolló un conjunto de herramientas de evaluación de software con las que se podrá analizar y comparar la calidad del código de fuente y probar su idoneidad para su despliegue empresarial. El coste total del proyecto se estima en unos 2.470 millones de euros.

- Estas herramientas sólo estimarán la calidad del producto.

Modelo de QSOS

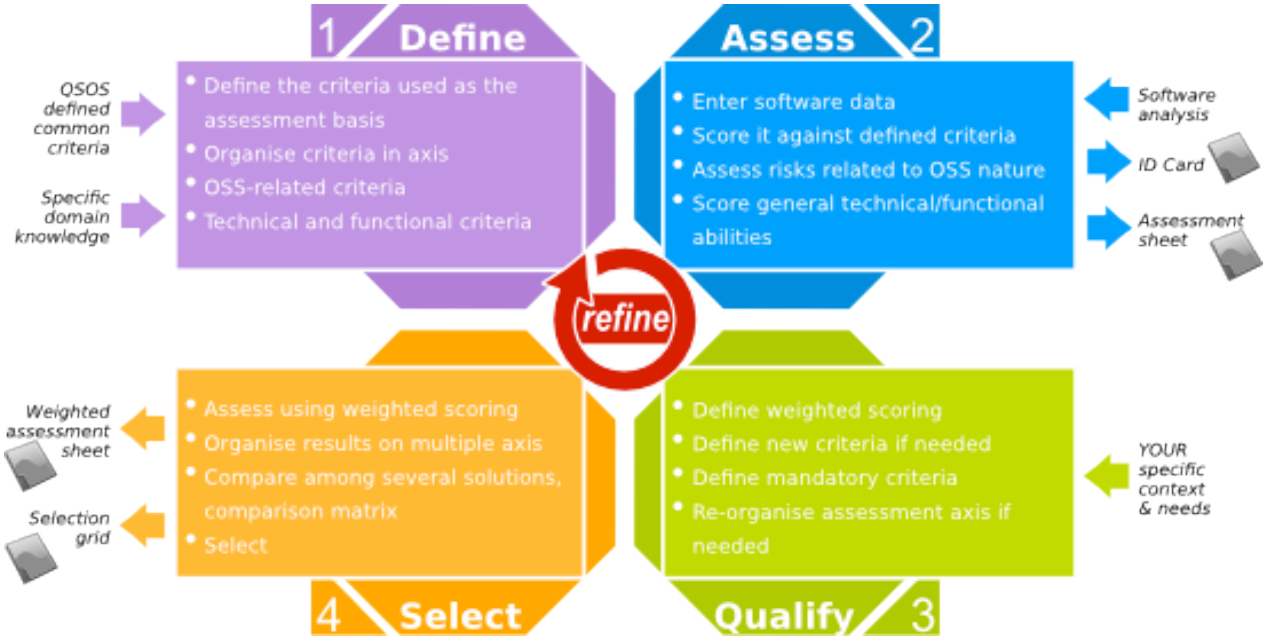
- Uno de los modelos que permite la cuantificación y calificación de software Open Source es el Method for Qualification and Selection of Open Source Software (QSOS).

- Está orientado exclusivamente al producto de software.

- Más información en http://www.qsos.org/

Metodología del Modelo QSOS

- Es un proceso que consiste en 4 pasos que pueden ser refinados. A saber:



Pasos de la Metodología

1. **Definición:**
   Constitución y enriquecimiento de los marcos de referencia que serán utilizados en los pasos siguientes.

2. **Evaluación:**
   Evaluación del software hecho de acuerdo a 3 ejes de criterios: cobertura funcional, riesgos del usuario y riesgos del proveedor de servicios (independientemente de cada usuario particular/ contexto de uso).

3. **Calificación:**
   Carga de los criterios divididos en 3 ejes, modelando el contexto (requerimientos de usuario y/o estrategia escogida por el proveedor de servicios).

4. **Selección:**
   Aplicación del filtro configurado en el paso anterior a los datos encontrados en los dos primeros pasos, de manera de realizar consultas, comparaciones y selección de productos.

Paso 1 : Definición

- El objetivo de este paso es definir varios elementos de la tipología a ser utilizada por los 3 pasos que siguen. Los marcos de referencia son:

1. **Familia de Software.**
   Este aspecto responde la pregunta "¿Qué tipo de software estamos analizando?".

2. **Tipos de Licencia.**
   Clasificación de las licencias más comunes de Software Libre y de código abierto.

3. **Tipos de comunidades.**
   Clasificación de las comunidades que pueden desarrollar Software Libre u Open Source.

Paso 2 : Evaluación

- Este paso tiene como objetivo la colección de información por parte de las comunidades de código abierto. Esta evaluación comprende la elaboración de la tarjeta de identificación del software, así como la elaboración de la hoja de evaluación del software.

- La **tarjeta de identificación del software** contiene datos y hechos acerca del software, es utilizada como base para el proceso de evaluación. Contiene elementos como nombre, fechas de creación, tipo de software autores, descripción general, los servicios que presenta, aspectos técnicos y funcionales, entre otros.

- Por otra parte la **hoja de evaluación**, contempla la identificación, descripción y análisis en detalle de cada versión que se presenta del software.

Paso 2 : Evaluación

- La tarjeta de identificación cubre lo siguiente:
  - **Información general.**
    - Nombre del software
    - Referencia, fecha de creación, fecha de elaboración de esta tarjeta
    - Autor
    - Tipo de software
    - Entre otros…
  - **Servicios existentes.**
  - Documentación
  - Entre otros…
  - **Aspectos técnicos y funcionales.**
  - Tecnologías de implementación
  - Funcionalidades detalladas
  - **Síntesis y comentarios generales.**

Paso 2 : Evaluación

- La hoja de evaluación cubre lo siguiente:

- **Puntaje** que va del 0 al 2 y que son establecidos durante el paso de Calificación dependiendo de los requerimientos del usuario.

- **Cobertura funcional** determinada por la definición establecida en el paso de Definición.

- **Riesgos desde la perspectiva del usuario** a los que está expuesto cuando escoge una solución de Software Libre u Open Source.

- **Riesgos desde la perspectiva de un proveedor de servicios** que utilice dicha solución de software.

Paso 3 : Calificación

- El objetivo de este paso es definir los filtros que traduzcan las necesidades y restricciones relacionadas con la selección del software de código abierto en un contexto especifico. Para ello se definen niveles de filtros sobre el software en base:

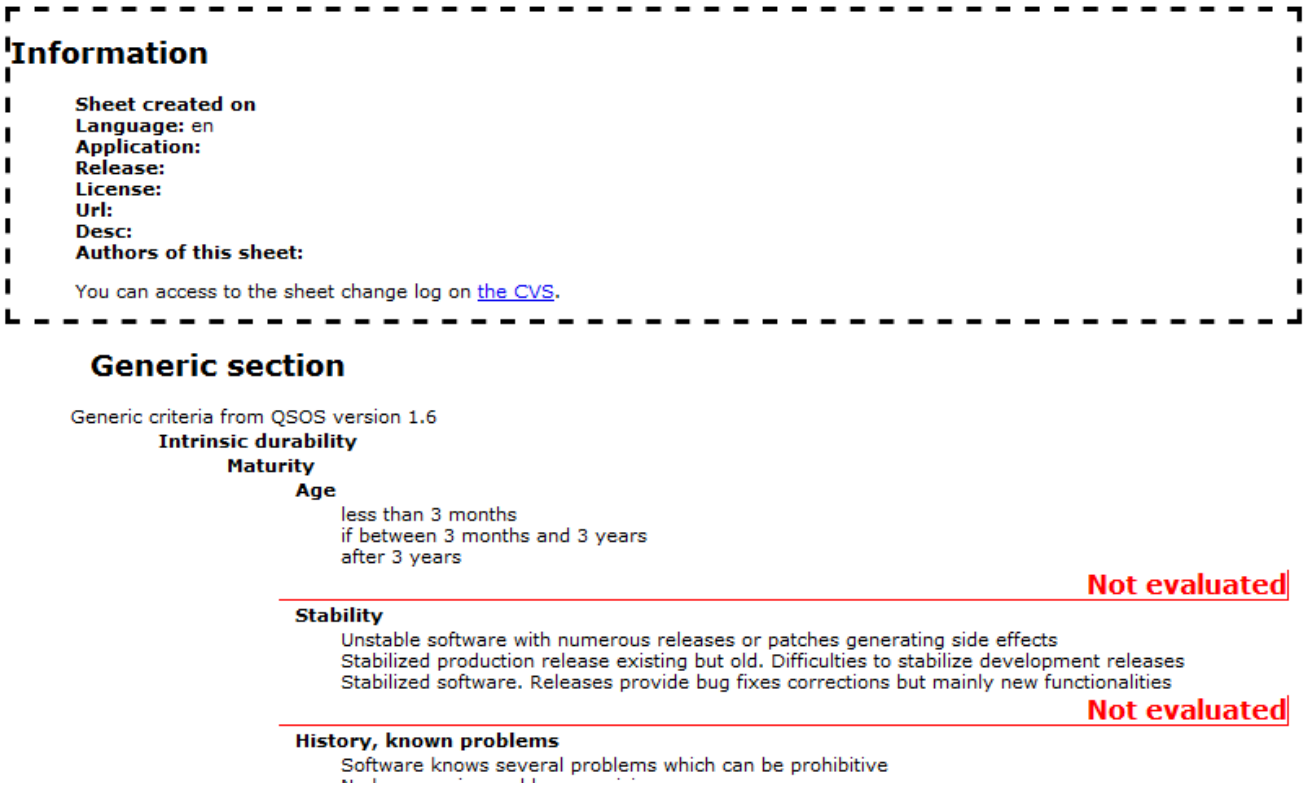
- Tenemos cuatro tipos de filtros:
  - Filtros sobre la tarjeta de **identificación**.
  - Filtros sobre las **funcionalidades**.
  - Filtros sobre los **riesgos desde la perspectiva del usuario**.
  - Filtros sobre los **riesgos desde la perspectiva del proveedor de servicios**.

Paso 4 : Selección

- Este paso tiene como objetivo identificar el software que contenga y satisfaga los requerimientos de usuario, o de manera más general permita la comparación de software de una misma familia. Puede ser de dos modos: un modo estricto (selección estricta), y otro un poco más holgado (selección holgada).
- La **selección estricta** se basa en la eliminación del software tan pronto como el software no cumpla con lo formulado en el paso de Calificación. Este método es muy restrictivo y puede no seleccionar software alguno.
- La **selección holgada** se basa en darle puntuación nuevamente al software dependiendo de lo obtenido en el paso de Evaluación. Al final se escoge el software con más (o menos) puntos.

Paso 4 : Selección

- Así luce una plantilla de una hoja de evaluación de QSOS:



Information

Sheet created on
Language: en
Application:
Release:
License:
Url:
Desc:
Authors of this sheet:

You can access to the sheet change log on the CVS.

Generic section

Generic criteria from QSOS version 1.6

Intrinsic durability

Maturity

Age

less than 3 months
if between 3 months and 3 years
after 3 years

Not evaluated

Stability

Unstable software with numerous releases or patches generating side effects
Stabilized production release existing but old. Difficulties to stabilize development releases
Stabilized software. Releases provide bug fixes corrections but mainly new functionalities

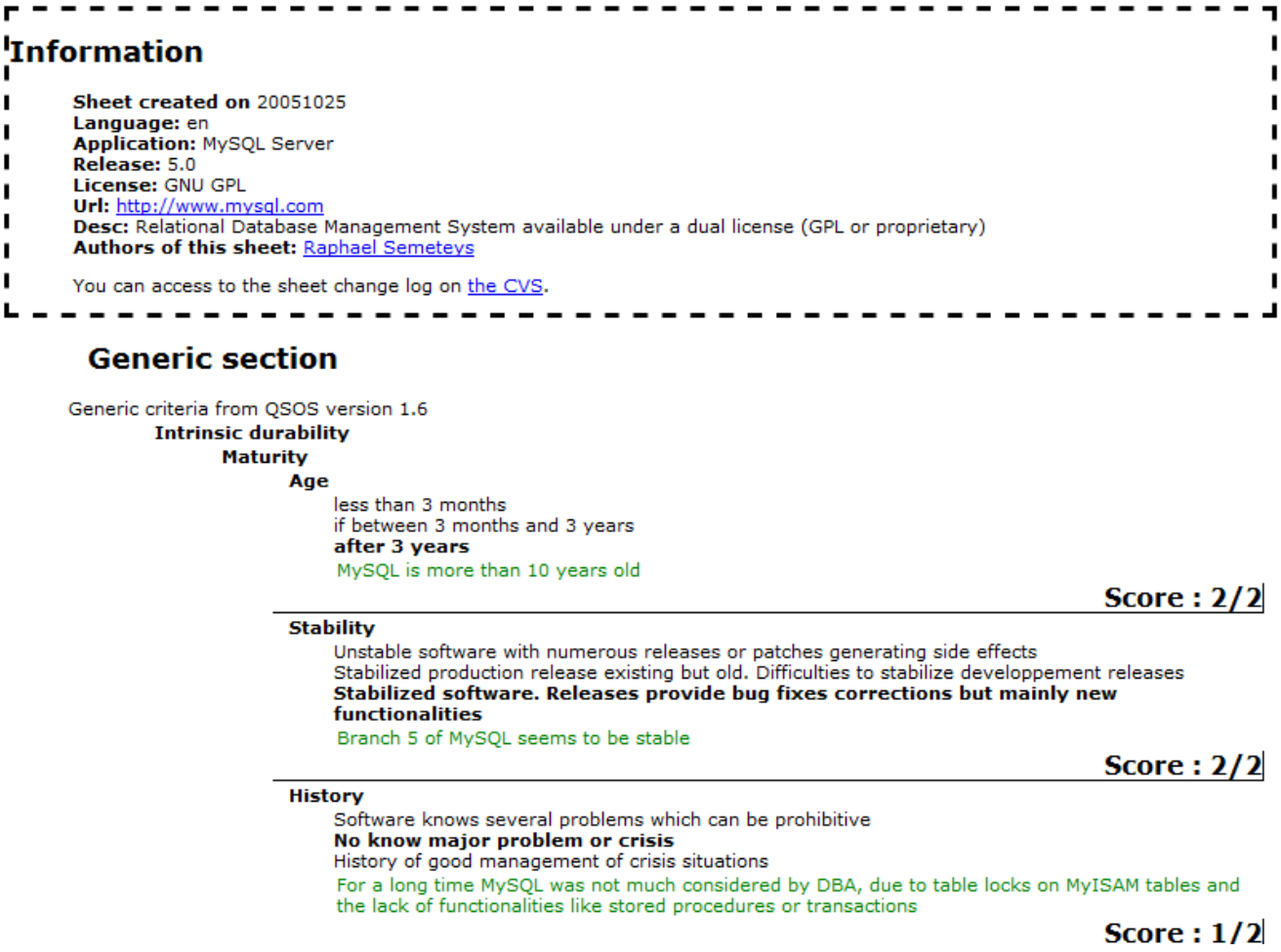
Not evaluated

History, known problems

Software knows several problems which can be prohibitive

Paso 4 : Selección

- Así luce una hoja de evaluación de QSOS:



Information

Sheet created on 20051025
Language: en
Application: MySQL Server
Release: 5.0
License: GNU GPL
Url: http://www.mysql.com
Desc: Relational Database Management System available under a dual license (GPL or proprietary)
Authors of this sheet: Raphael Semeteys

You can access to the sheet change log on the CVS.

Generic section

Generic criteria from QSOS version 1.6

Intrinsic durability

Maturity

Age

less than 3 months
if between 3 months and 3 years
after 3 years
MySQL is more than 10 years old

Score : 2/2

Stability

Unstable software with numerous releases or patches generating side effects
Stabilized production release existing but old. Difficulties to stabilize developpement releases
Stabilized software. Releases provide bug fixes corrections but mainly new functionalities
Branch 5 of MySQL seems to be stable

Score : 2/2

History

Software knows several problems which can be prohibitive
No know major problem or crisis
History of good management of crisis situations
For a long time MySQL was not much considered by DBA, due to table locks on MyISAM tables and the lack of functionalities like stored procedures or transactions

Score : 1/2

Métricas usadas por QSOS

- Básicamente la metodología busca dar indicadores sobre la funcionalidad que presta un software determinado y los riesgos que podría corre un usuario y/o un proveedor de servicios con dicho software.

- Para obtener esos indicadores QSOS utiliza dos tipos de métricas:
- **Métricas generales**: que se aplican a todo tipo de Software Libre u Open Source.
- **Métricas específicas**: que se aplican a una familia determinada de software.
  - Las **métricas generales** se describen en la "*Generic Section*" de la hoja de evaluación y se encuentran justo debajo de la tarjeta de identificación. Este tipo de métrica comprende aspectos como madurez, actividad en el desarrollo, portabilidad, entre otras.
  - Las **métricas especificas** se describen justo después de la "*Generic Section*". Comprenden aspectos inherentes a las características del tipo de software. Por ejemplo, para la familia de software de RDBMS se contempla el soporte de SQL, el soporte de *constraints* sobre las tablas, entre otros.

Métricas usadas por QSOS

- **Durabilidad intrínseca (sustentabilidad)**
  - **Madurez**
    - Edad
    - Estabilidad
    - Historia, problemas conocidos
    - Probabilidad de forks, fuentes del forking
  - **Adopción**
    - Popularidad (relacionada con: público en general, expertos, ...)
    - Referencias (si se emplea en alguna solución conocida)
    - Comunidad de contribuyentes (nivel de actividad)
    - Libros disponibles
- **Liderazgo de desarrollo**
  - Equipo de desarrollo (tamaño)
  - Estilo de gerencia (“dictatorial”, “un poco déspota”, “consejo de arquitectos”)
- **Actividad**
  - Desarrolladores (número total de desarrolladores, cargos bien /mal definidos e identificados)
  - Actividad en solución de problemas
  - Actividad en el desarrollo de funcionalidades
  - Actividad en nuevos lanzamientos

Métricas usadas por QSOS

- **Solución industrializada**
  - **Independencia del desarrollo** (si el software es desarrollado por una única compañía)
  - **Servicios**
    - Entrenamiento
    - Soporte
    - Consultoría
  - **Documentación** (no disponible, disponible /actualizada, disponible/no actualizada)
- **Aseguramiento de la calidad**
  - Aseguramiento de la calidad (utilizando algún método o modelo reconocido)
  - Herramientas (feedback u alguna otra herramienta que monitoree el progreso)
- **Empaquetamiento** (paquete oficial, ofrecido por la comunidad, no disponible)
  - Fuente
  - Debian
  - FreeBSD
  - HP-UX
  - MacOSX
  - Mandriva

Métricas usadas por QSOS

- **Explotabilidad**
  - **Facilidad de uso, ergonomía** (si requiere de conocimientos técnicos: bajo, medio o alto)
  - **Administración / Monitoreo** (si proporciona herramientas de administración/monitoreo)
- **Adaptabilidad técnica** (inherente al código fuente)
  - **Modularidad** (software: monolítico, modularidad de primer nivel, completamente modular)
  - **Modificación del código** (compilación: difícil y a mano, posible y a mano,…)
  - **Extensión del código** (requiere re-compilación, uso y manejo de plugins)
- **Estrategia**
  - **Licencia**
    - Permisividad (sólo si el usuario quiere hacerse dueño del código)
    - Protección respecto a forks propietarios
  - **Propietario de los copyrights** (si es un individual, una comunidad o una empresa)
  - **Modificación del código fuente** (imposible, uso de repositorios, …)
- **Roadmap** (públicado, no publicado)
- **Patrocinante**
- **Independencia estratégica**

Caso de Estudio : QSOS

- Supongamos una empresa que está evaluando la posibilidad de incluir alguna de tres aplicaciones de software conocidas de bases de datos relacionales: MaxDB, MySQL y PostgreSQL en una aplicación web propia.
- Para ello utilizó una herramienta web disponible en el website de QSOS que permite aplicar los dos últimos pasos de la metodología sobre unas hojas de evaluación previamente definidas y que están disponibles en el website. Dicha herramienta se denomina O3S.

- En un **primer paso**, la herramienta solicita al usuario que especifique la familia de software a evaluar.

Step 1 - Select a software family

Software families
BI-Suite
RDBMS
bugTracker
cms
dbtools
esb
etl
forge
groupware

Caso de Estudio : QSOS

- **Luego**, aparece una tabla donde el usuario podrá especificar los pesos que le asigna a cada aspecto del software (de acuerdo con la familia elegida en el paso anterior). Estos pesos serán utilizados para calcular el puntaje final y ver qué software se ajusta más a las necesidades del usuario.

Step 2 - Set your weightings

< Step 1 | Save | Step 3 >

Choose File | No file chosen | Upload

RDBMS	Weight
⊟ Generic section	1
⊟ Intrinsic durability	1
⊟ Maturity	1
Age	2
Stability	1
History	3
Fork	2
⊟ Adoption	1
Popularity	3
References	3
Contributing Community	2
books	2

Caso de Estudio : QSOS

- El **tercer paso** consiste en elegir si se quiere mostrar los resultados vía web, en un documento QSOS XML o en un documento de OpenOffice.org.

Step 3 - Select software

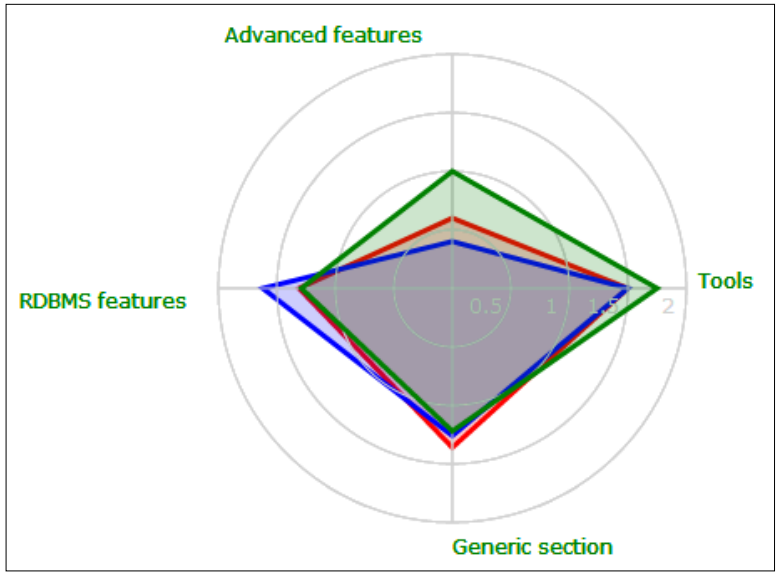
< Step 2

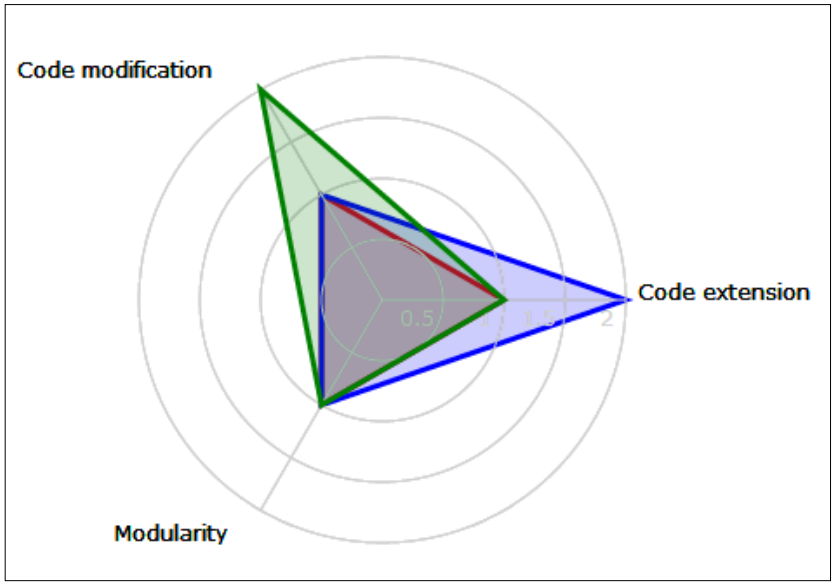
RDBMS	QSOS XML File	OpenDocument	Step 4 - View/Compare
mysql			
mysql-5.0	<?xml		☐
mysql-4.1	<?xml		☐
postgresql			
postgresql-8.0	<?xml		☐
maxdb			
maxdb-7.6	<?xml		☐

My browser supports SVG ☑

Caso de Estudio : QSOS

- Es posible ver un gráfico de radar (en los resultados) donde se pueden establecer comparaciones entre el software elegido.

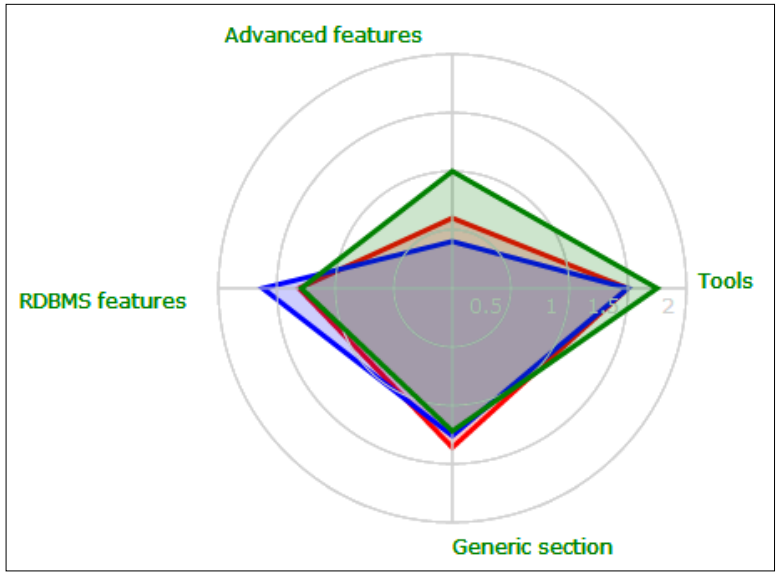


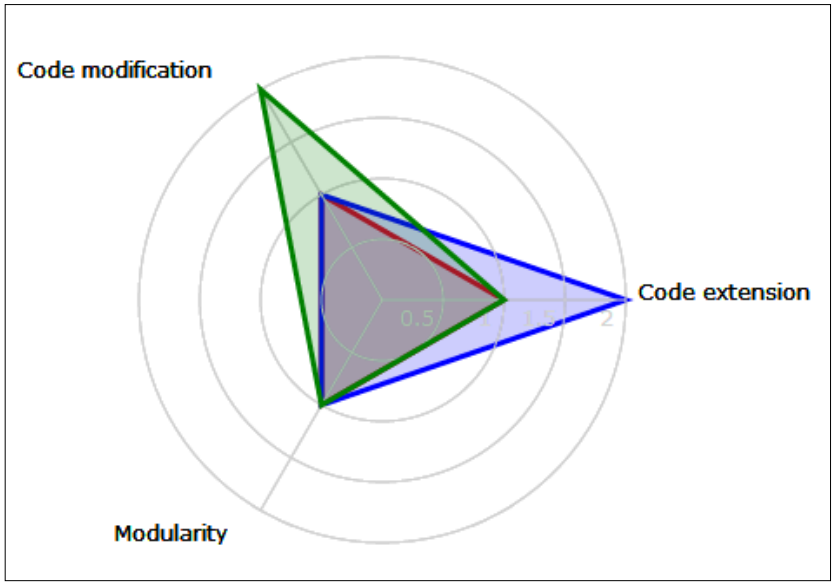


MySQL Server 5.0 PostgreSQL 8.0 Max DB 7.6

Caso de Estudio : QSOS

- La empresa seleccionó MaxDB ya que ofrece más características avanzadas y facilita la modificación del código.





MySQL Server 5.0 PostgreSQL 8.0 Max DB 7.6

Caso de Estudio : MOSCA

- Dos empresas están desarrollando una aplicación web cada una. Quieren conocer qué tan bueno es su software. Para ello buscaron ayuda del LISI.

- Se aplicaron dos tipos de cuestionarios.

- El **primero** enfocado al producto y dirigido a tres grupos de evaluación: el Líder del Proyecto, Desarrolladores-Analistas y Usuarios.

- El **segundo** tipo de cuestionario está enfocado al proceso de desarrollo y va dirigido a dos grupos de evaluación: el Líder del Proyecto y los Desarrolladores-Analistas.

Caso de Estudio : MOSCA

- Los datos de las dos empresas fueron analizados tomando en cuenta:
  - Las **categorías del producto** seleccionadas por la empresa junto con el evaluador
  - Las **categorías del proceso**
  - Las **características del producto y del proceso**.
- En primer lugar, se analizan los datos referentes al producto.
  - Las categorías seleccionadas (aparte de Funcionalidad) por ambas empresas fueron: Mantenibilidad y Usabilidad.
  - Se debe recordar que según el algoritmo del modelo MOSCA, la empresa debe seleccionar exactamente 2 categorías que identifiquen a su producto de software.

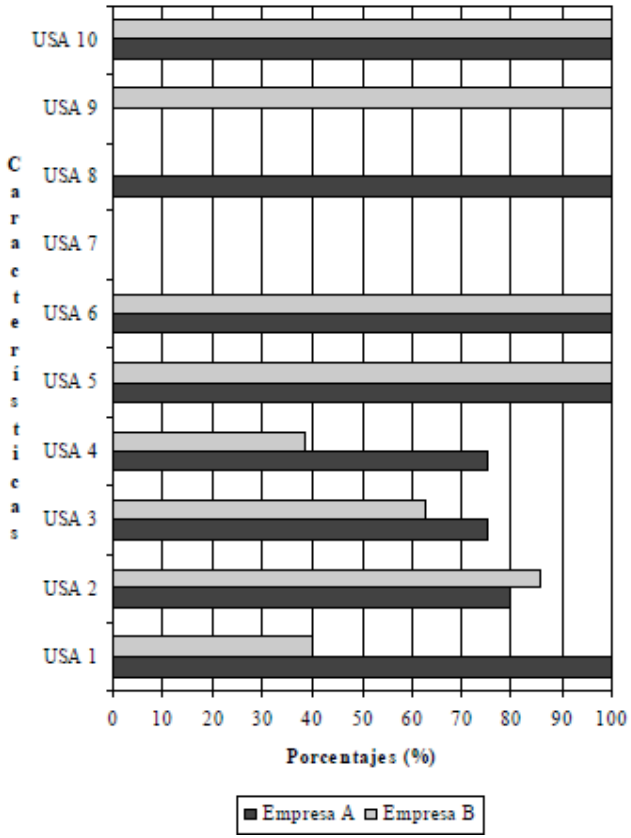
Caso de Estudio : MOSCA

- La **Empresa A** seleccionó la categoría **usabilidad**, ya que el sitio Web debe ser un producto atractivo, entendible y fácil de utilizar para los usuarios del mismo. Lo más importante de esta aplicación es su *front-end*, por lo cual el mismo debe cumplir los requerimientos de la categoría Usabilidad. La otra categoría seleccionada fue **mantenibilidad**, ya que el producto debe ser actualizado constantemente y por ello debe tener la capacidad de ser modificado sin ningún problema.

- La **Empresa B** seleccionó la categoría **usabilidad**, ya que su producto está destinado a diferentes tipos de usuarios y la dificultad en el uso del mismo debe ser mínima. Además, esta aplicación debe ser atractiva, ya que el éxito de la misma, dependerá del grado de satisfacción de los usuarios. La otra categoría seleccionada fue **mantenibilidad**, ya que el producto de software está en constante desarrollo y debe ser capaz de aceptar cualquier tipo de modificaciones.

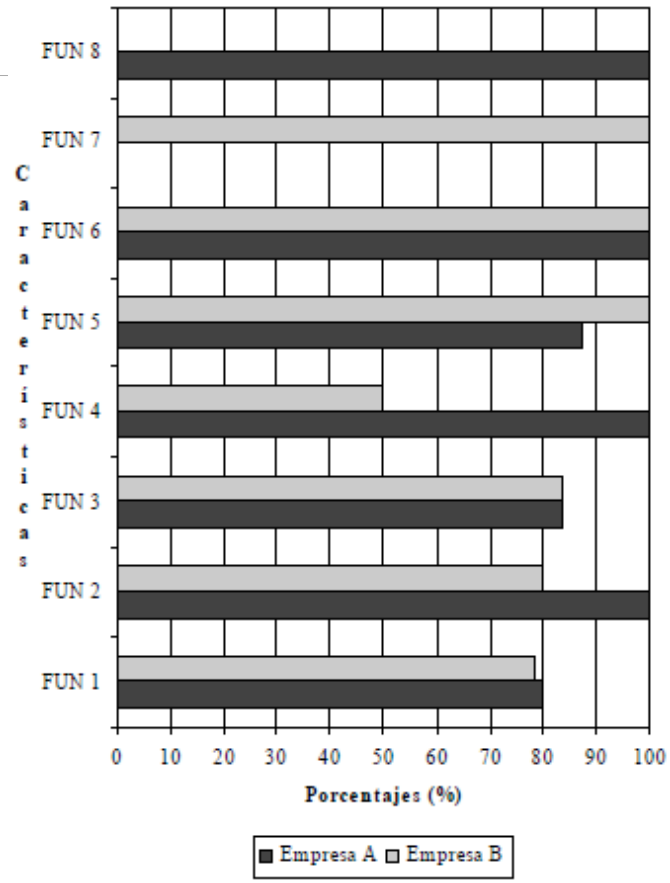
Caso de Estudio : MOSCA

Porcentajes de satisfacción de los productos frente a la característica **USABILIDAD**.



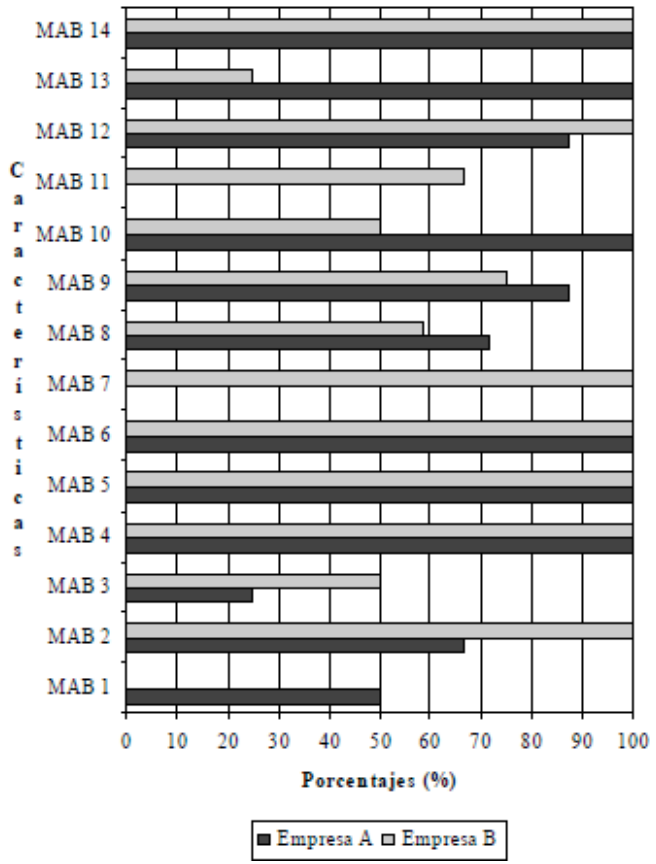
Caso de Estudio : MOSCA

Porcentajes de satisfacción de los productos frente a la característica **FUNCIONALIDAD**.



Caso de Estudio : MOSCA

Porcentajes de satisfacción de los productos frente a la característica **MANTENIBILIDAD**.



Fuentes Consultadas

- http://prof.usb.ve/lmendoza/Documentos/PS-6116/Guia%20Arquitectura%20v.2.pdf
- http://books.google.co.ve/books?id=DR74RkJlBTMC&printsec=frontcover&dq=la+calidad+del+software+y+su+medida&ei=CYzGSdG8LjjSzATF_ZjaDQ#PPA12,M1
- http://www.monografias.com/trabajos55/proceso-de-desarrollo-software/proceso-dedesarrollo-software2.shtml
- http://www.ub.edu.ar/catedras/ingenieria/ing_software/ubftecwwwdfd/calidadsw/criterios.htm
- http://eisc.univalle.edu.co/materias/Material_Desarrollo_Software/Metricas4.pdf
- http://www.ejournal.unam.mx/cys/vol08-03/CYS08304.pdf . Anna Grimán.
- http://www.qsos.org

Gracias por su atención
Sesión de preguntas y comentarios