UNIVERSIDAD NACIONAL DE LA PLATA
FACULTAD DE INFORMÁTICA

CLASIFICACIÓN DE HERRAMIENTAS

OPEN SOURCE PARA PRUEBAS DE APLICACIONES WEB

UN CASO DE ESTUDIO

TESINA DE GRADO

AUTORAS: RODRIGUEZ, ANAHI S. | SORIA, VALERIA
DIRECTOR: LIC. DÍAZ, F. JAVIER | CODIRECTOR: LIC. BANCHOFF T., CLAUDIA M.

Estructura de la presentación

* Objetivo y Motivación.
* Pruebas de Software en el proceso de desarrollo en Software Libre.
* Etapa de pruebas de software.
* Pruebas de software.
* Criterios de evaluación de herramientas FLOSS para pruebas de software.
* Plantilla Propuesta.
* Presentación del video.
* Conclusiones.
* Trabajo a futuro.

Objetivo y Motivación

Objetivo y Motivación

La etapa de pruebas es la menos sistematizada y tenida en cuenta.

Existe una gran cantidad de parches y versiones surgidas luego del lanzamiento de una versión final de un software, destinadas a cubrir "agujeros de seguridad" o malos funcionamientos.

Es por esto que es importante realizar pruebas en el software.

Objetivo y Motivación

Ante la gran diversidad y variedad de de herramientas Open Source es necesario contar con un instrumento que nos ayude a seleccionarlas de manera metodológica y objetiva.

Se analizaron modelos de madurez de software y se construyó una planilla que sintetiza los principales criterios de evaluación de madurez.

Objetivo y Motivación

En este trabajo se recopila información (dispersa) de herramientas de testing, se agrupa y se unifica la terminología para describir a las mismas.

Para ilustrar la validez del trabajo realizado se incluye una prueba de concepto sobre una aplicación Web.

Pruebas de Software en el proceso de desarrollo en Software Libre

Pruebas de Software en el proceso de desarrollo en Software Libre

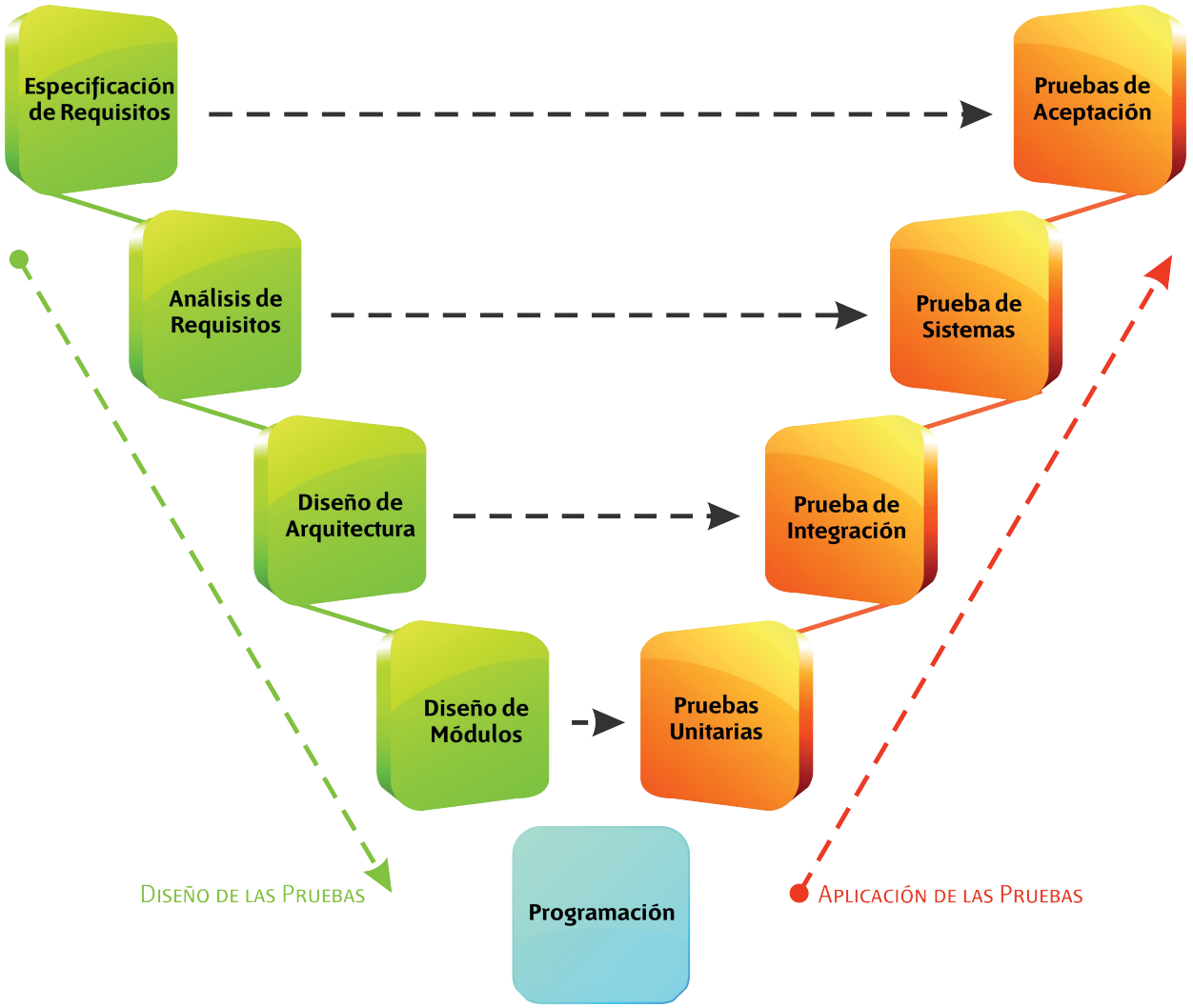
Existen distintos modelos para el desarrollo de software.

Cada uno de los modelos organiza las distintas tareas involucradas de manera diferente según la política del mismo.

Software Libre (SL): compartir el trabajo.

Modelo en V

Modelo en V


Especificación de Requisitos
Pruebas de Aceptación
Análisis de Requisitos
Prueba de Sistemas
Diseño de Arquitectura
Prueba de Integración
Diseño de Módulos
Pruebas Unitarias
Programación
Diseño de las Pruebas
Aplicación de las Pruebas

Licencias de Software

Una Licencia de Software es la autorización o permiso concedida por el autor para utilizar su obra de una forma convenida habiendo marcado unos límites y derechos respecto a su uso.

GNU GPL es una licencia copyleft y libre para software y otros tipos de obras.

Garantiza la libertad de compartir y modificar todas las versiones de un programa, y asegura que siga siendo software libre para todos sus usuarios.

Tabla comparativa de licencias con GNU GPL

Licencia	Ultima Versión	Copyleft	Compatible GNU GPL
Apache	2.0	✗	✓
Artistic License	2.0	✗	✓
BSD	S/D	✗	✓
CPL (Common Public License)	1.0	✓	✗
GNU GPL	V3	✓	✓
GNU LGPL (Lesser LGL)	V3	✓	✓
MPL	2.0	✓	✓
MIT license	S/D	✗	✓
Python License (CNRI Python License)	S/D	✓	✗
W3C Software License	S/D	✗	✓

Etapa de pruebas de software

"El testing puede ser usado para mostrar la presencia de bugs,
pero nunca para mostrar su ausencia".
Edsger Dijkstra

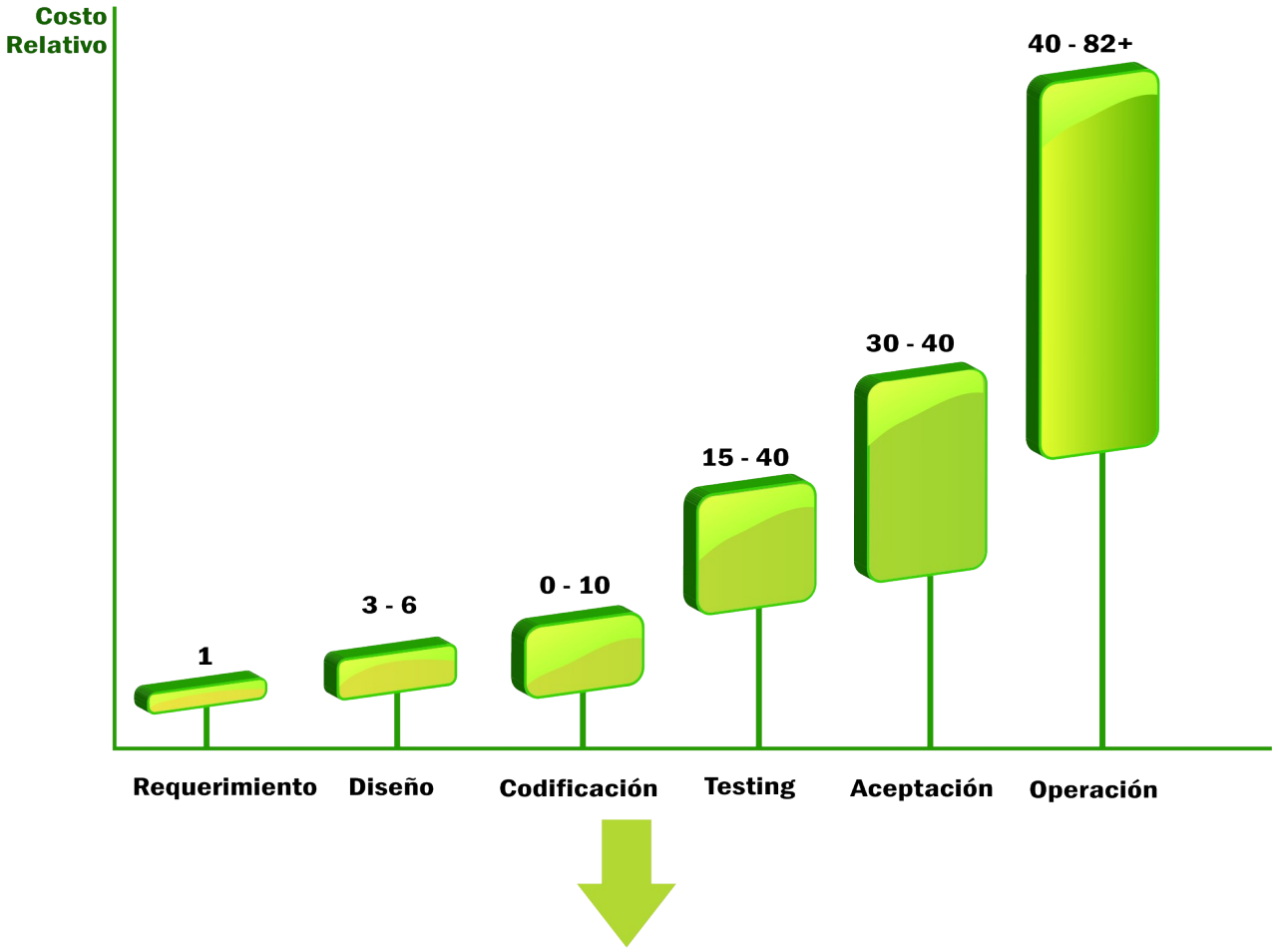
Etapa de pruebas de software

Probar un software es someterlo a ciertas condiciones que puedan demostrar si es válido o no con respecto a los requerimientos planteados.

El objetivo de las pruebas es encontrar la mayor cantidad de errores posibles, preferentemente antes de que los encuentre el usuario final.

El testing mejora la calidad del producto y deber realizarse en cada una de las etapas del desarrollo del software.

Beneficios de la corrección de errores en etapas tempranas

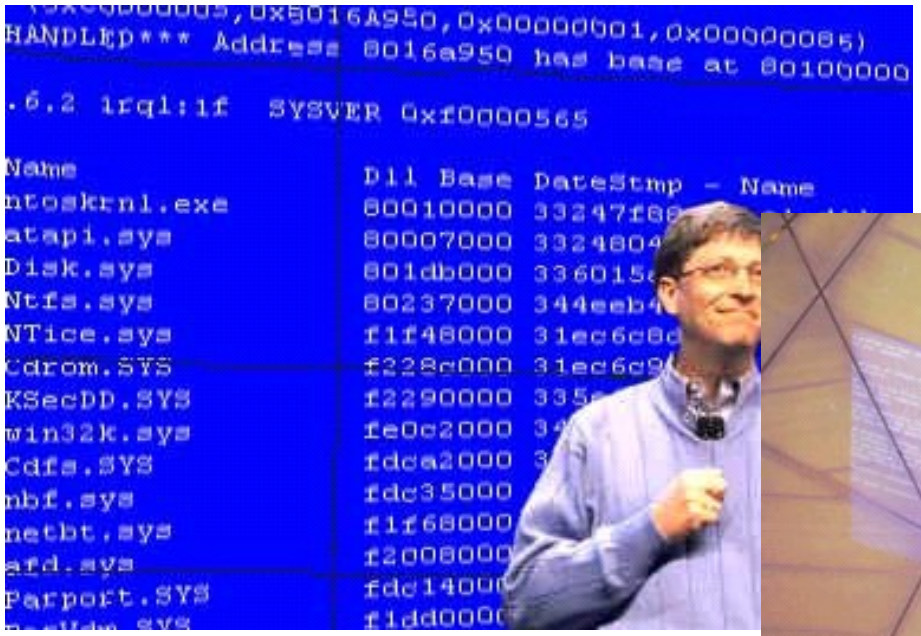


Cuanto más temprano se inicie el testing en el proceso de desarrollo de software, mayor será su efectividad

¿Para qué sirve probar?

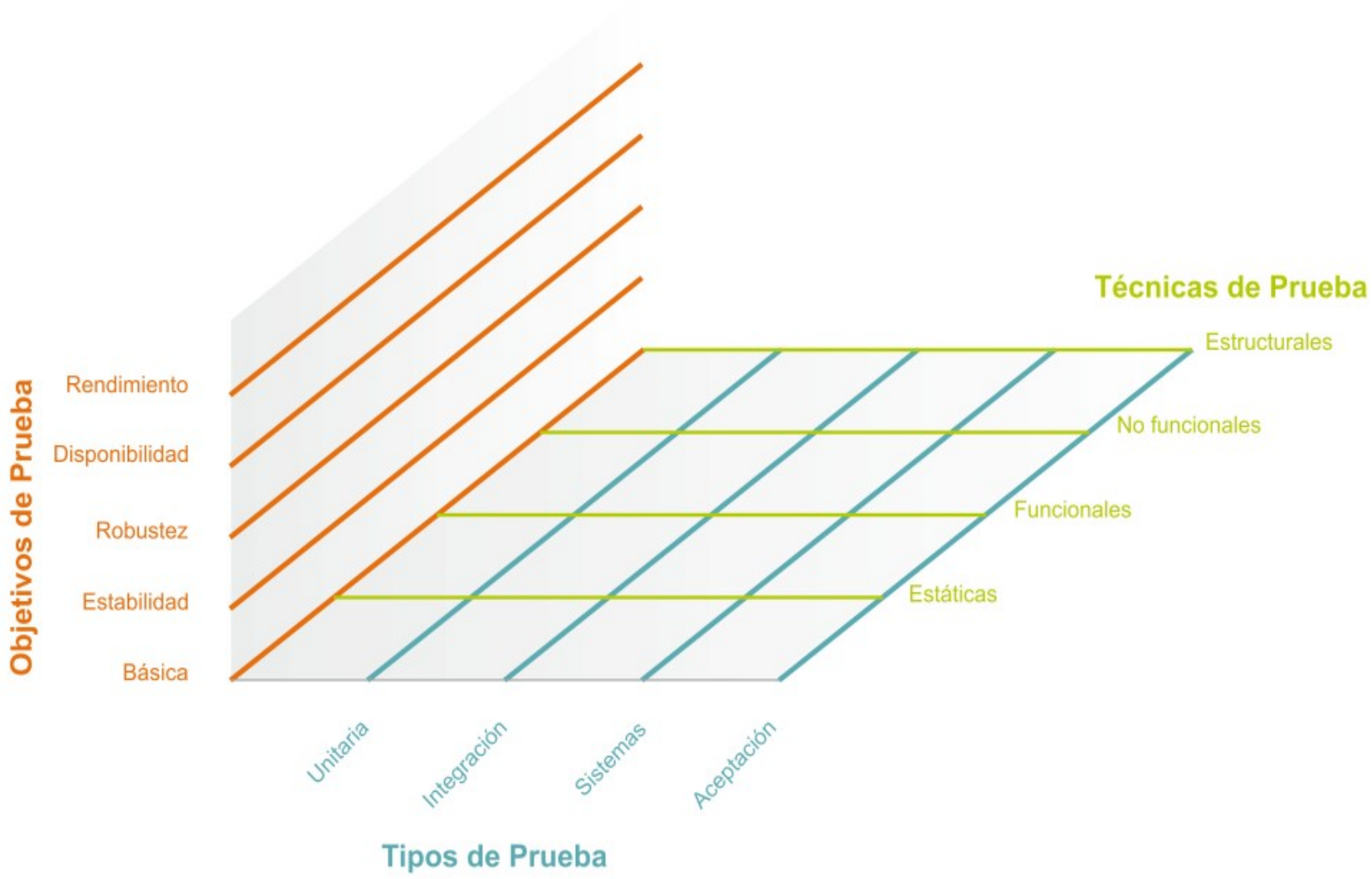
* División en coma flotante en Intel Pentium (1994)
* Desintegración del Ariane 5 (1996)
* Bugs en software bancario (1996)
* Error en equipo de Cisco (1998)
* Error del milenio (2000)
* Fallo de seguridad en las cámaras IP de TRENDnet (2012)

¿Para qué sirve probar?



Pruebas de software

Eje del trabajo de testing



Caso de prueba

Objetivo: la característica del sistema a probar.

Datos de entrada y de ambiente: datos a introducir al sistema que se encuentra en condiciones preestablecidas.

Comportamiento esperado: la salida o la acción esperada en el sistema de acuerdo a los requerimientos del mismo.

Comprobación del resultado esperado: método o forma de realizarlo.

Ambiente de pruebas

Todo el hardware necesario para la prueba: Procesadores, periféricos, drivers, red, etc.

Todo el software necesario para las pruebas: Sistema Operativo, Software de BD, Software a probar, etc.

Interfaz en general: Archivos recibidos de otros sistemas, etc.

Automatización de las pruebas

Beneficios en sus tareas con la utilización de herramientas.

Ayudar a realizar el trabajo repetitivo y automático, facilitar las tareas de regresión, comparación de grandes volúmenes de datos, simulación de comportamiento.

El uso de una herramienta no garantiza el éxito en las pruebas.

Los resultados deben ser analizados e interpretados por un especialista.

Criterios de evaluación de herramientas FLOSS para pruebas de software

Métodos para evaluar la madurez de los proyectos Open Source

¿Cuál es la durabilidad del software?

¿Qué nivel de estabilidad tiene o se espera?

¿Es fácil y tangible agregarle funcionalidad al software?

¿Tiene una comunidad activa que lo avale?

Métodos para evaluar la madurez de los proyectos Open Source

OSMM (Open Source Maturity Model de Capgemini)

Define 27 parámetros divididos en las siguientes categorías: Producto, Integración, Uso, Aceptación, Necesidades del Usuario. Cada uno de estos parámetros puede tener un puntaje entre 1 (menos importante) y 5 (más importante).

OSMM (Open Source Maturity Model de Navica)

Se definen 6 características. El proceso sigue un ciclo de 4 fases: selección del software, ponderación de la categoría, aplicación en las planillas y calculo de la puntuación final (puntuación de cada categoría * ponderación).

Métodos para evaluar la madurez de los proyectos Open Source

BRR (Business Readiness Rating)

Este proceso sigue 4 pasos: se crea una lista con el software a evaluar, se clasifican y ponderan los criterios de selección, se recopilan los datos para cada criterio y se realiza el cálculo y publicación de los resultados.

Qualification and Selection of Open Source Software (QSOS)

El proceso consiste en 4 pasos independientes entre si e iterativos: Definición, Evaluación, Calificación y Selección.

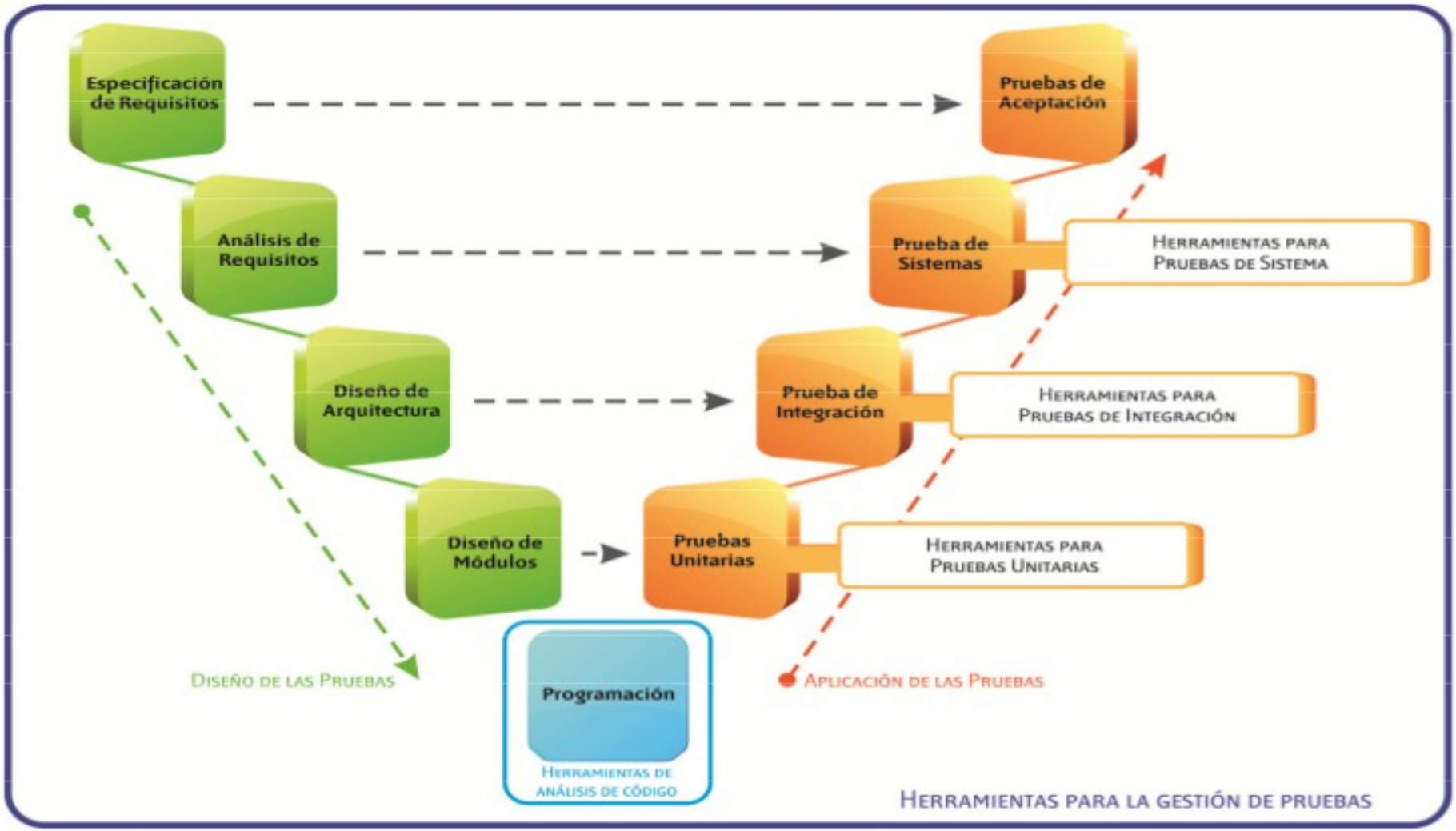
Métodos para evaluar la madurez de los proyectos Open Source

NASA's Reuse Readiness Levels (RRL)

Se identificaron 9 áreas temáticas como ser Documentación, Extensibilidad, Cuestiones de Propiedad Intelectual, Modularidad, Empaquetado, Portabilidad, Cumplimiento de estándares, Soporte y Verificación y Pruebas.

Existen 9 niveles de RRL que van de 1 (menos maduro) a 9 (más maduro).

Clasificación de Herramientas



Plantilla propuesta

Plantilla propuesta

Descripción general: una breve descripción de la funcionalidad de la herramienta, mencionando la URL del sitio oficial, su última versión, el lenguaje en el que está desarrollada, etc.

Documentación: el tipo de documentación asociada existente. Se debe encontrar en el sitio oficial o dentro del producto.

Madurez: Se relevarán datos referidos a la herramienta que permitan medir el nivel de madurez del software.

Licencia: se nombrará el tipo de licencia a la que pertenece la herramienta.

Plataforma: se nombrará el Sistema Operativo (SO) en el que pueden ser instaladas dichas herramientas (portabilidad).

Interfaz: se nombrará el tipo de interfaz (GUI o consola) con el que cuenta la herramienta.

Datos generales

Herramienta	Versión	Fecha de Inicio del proyecto	Licencia	Plataforma	Tipo de Interfaz	Lenguaje
PHPUnit	V 3.6 Diciembre 2011	S/D	GNU GPL	Independiente	Consola	PHP
JMeter	V 2.6 Enero 2012	Diciembre 1998	Apache License V 2.0	Independiente	GUI	JAVA

Criterios de documentación

Herramienta	Guía de instalación	Manual de usuario	Preguntas Frecuentes	Soporte Online			Adicional
				Foro	Lista de mail	Blog	
PHPUnit	SI	SI	NO	NO	SI	NO	S/D
JMeter	NO	SI	SI	SI	SI	NO	Wiki JavaDocs (API)

Criterios de madurez (*)

Herramienta	Inicio del proyecto*1	Grado de actualización*2	Actividad en lanzamientos*3	Actividad en el reporte de errores*4
PHPUnit	S/D	MB	MB	MB (Último movimiento marzo 2012)
JMeter	B	B	B	MB (último movimiento marzo 2012)

(*) Se completa con las siguientes siglas:

M = Malo B = Bueno

R = Regular MB = Muy bueno

*1 Se refiere a que si es muy joven quizás no tenga una madurez suficiente

*2 La última versión se encuentra cercana al año actual

*3 Se refiere a la frecuencia en la publicación de versiones

*4 Se refiere a la frecuencia con que se resuelven y reportan errores

Presentación del video

Conclusiones

Conclusiones

En necesario seleccionar una herramienta que se ajuste a las necesidades de cada proyecto en base a su madurez.

Es por esto que se propone una plantilla que sintetiza las principales características para medir la madurez de un software.

En este trabajo se recopila información (dispersa) de herramientas de testing, se agrupa y se unifica la terminología para describirlas a las mismas.

Se planteó un caso de estudio concreto sobre un sistema Web desarrollado por el CeSPI.

Se realizaron pruebas de unidad utilizando PHPUnit y pruebas de rendimiento usando JMeter.

Conclusiones

Como resultado de esta tesina, debemos destacar que hemos podido establecer un modelo que nos permite seleccionar las herramientas y un procedimiento para la ejecución de pruebas de unidad y de rendimiento.

Trabajo a futuro

Trabajo a futuro

Revisar periódicamente las planillas, y de ser necesario agregar nuevas herramientas, dado el gran movimiento de la comunidad de Software Libre.

Seguir probando el sistema con diferentes herramientas.

Se podría extender la tesis agregando pruebas de usabilidad y accesibilidad, focos que no fueron tomados en este trabajo.

Se podría incluir el testing dentro de las materias de desarrollo de la carrera.

Preguntas ¿?

Muchas Gracias!