Solución de alta disponibilidad (HA) y balanceo de carga para el Servicio Web de la Red de Datos de la Universidad del Cauca

Jeimmy Viviana Cuellar Rivera
José Raul Romero Mera

ANEXO A

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Telecomunicaciones
Grupo I+D Nuevas Tecnologías en Telecomunicaciones
Popayán, 2012

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

ANEXO A METODOLOGÍA DE EVALUACIÓN DE CALIDAD OpenBRR

La metodología de evaluación Open *Business Readiness Rating*™ (OpenBRR) tiene como propósito permitir que toda la comunidad (adoptantes de las empresas, desarrolladores y usuarios) evalúen proyectos de software Libre o de Código Abierto de forma abierta y estandarizada, su proceso es descrito en el documento BRR 2005 - RFC 1 [1]. La evaluación de un proyecto de Software Libre o Código Abierto, FLOSS, mediante OpenBRR consiste en la asignación de valores ponderados a diversos factores del proyecto o software FLOSS y el cálculo de un puntaje que da la medida de idoneidad para su uso [2]. El modelo OpenBRR es un modelo abierto, flexible y estandarizado, esto permite la aplicación general de una evaluación sistemática y transparente, tanto para el software libre y/o de código abierto como para el software privativo.

El modelo OpenBRR se compone de un conjunto de 12 temas o categorías, cada una de la cuales contiene un conjunto de métricas. Estas categorías agrupan las diferentes dimensiones de calidad de una evaluación OpenBRR. Hay 27 métricas o indicadores únicos y genéricos a ser aplicadas en cada uno de los tipos de software para ser evaluados por el modelo. Además, las métricas o indicadores específicos de funcionalidad tienen que ser hechos a medida para cada tipo de software a ser evaluado [3].

La metodología consiste en 4 fases que son:

1. **Evaluación Rápida**: Creación de una lista del software candidato a ser evaluado.
2. **Evaluación Uso Objetivo**: Determinación de la importancia relativa de las categorías y las métricas.
3. **Recolección de datos y procesamiento**: Obtención manual de los datos para las métricas.
4. **Selección del aplicativo** - Selección del Proyecto o Aplicativo

Debe realizarse siempre la fase 1 primero, esto se hace mediante la identificación de uno o más candidatos de software, mientras que las fases 2 y 3 pueden ser realizadas en cualquier orden. Se trata de un proceso manual, y que aspira a ser completo, sencillo, adaptable y consistente.

Figura A. 1 Fases de OpenBRR

A.1 Fase 1: Evaluación Rápida:

El objetivo de esta fase es eliminar rápidamente los productos que son claramente inadecuados. Se identifican las mejores prácticas para hacerlo, que son las siguientes:

A. **Definir el uso objetivo de la aplicación (de misión crítica, desarrollo regular, o experimentación):** Dependiendo de cómo la aplicación está destinada a ser utilizada puede ser o no una opción viable. Por lo tanto, es pertinente evaluar el uso previsto del grupo de aplicaciones (su orientación funcional) antes de trasladarse a cualquier otro paso.

B. **Seleccionar un grupo de indicadores de viabilidad basado en el uso de destino**: Una de las desarrolladoras de esta metodología, la empresa SpikeSource, ha identificado varios indicadores de forma rápida y fácil de usar, que muestran claramente la viabilidad de un paquete de software para su adopción. Los elementos en la lista no son pertinentes para todos los usos de destino. Se debe seleccionar algunos indicadores de viabilidad que tienen más sentido para su uso objetivo. Estos indicadores pueden incluir:

- **Concesión de licencias / Legal:** No todas las licencias abiertas son creadas iguales por lo que algunas licencias son mucho más restrictivas que otras. Dos propiedades de licencia abierta son importantes en la adopción de software libre o de código abierto y por tanto deben tenerse en cuenta:
  - ✓ **Licencias *Open Source:*** También llamadas licencias aprobadas, son aquellas que están debidamente ratificadas por la *Open Source Initiative* (OSI) [4].
  - ✓ **Licencias *Copyleft***. Son licencias de software libre que permite la modificación de la base de código y la redistribución de la versión modificada, siempre y cuando el nuevo producto se mantenga abierto [5]. Las Licencias *copyleft* pueden suponer restricciones para los ISV (*Independent Software Vendor)*[1] que estén interesados en la integración de software de código abierto en un producto comercial.
- **Cumplimiento de los Estándares:** Si el cumplimiento de las normas de la industria es crucial para la categoría de software que se está evaluando, entonces se sugiere que el evaluador verifique que el *Software* cumpla con estas consideraciones antes de pasar a la evaluación completa de OpenBRR.
- **Referencias de Adoptadores:** Se tienen dos grupos de adoptadores, los entusiastas de ser los primeros en adoptar nuevas tecnologías o productos y aquellos que se siente cómodos en adoptar tecnologías o productos que otros ya están usando, con probabilidades de encontrar referencias que permitan cuantificar la satisfacción con los resultados hallados.
- **Disponibilidad de un soporte estable de una organización de apoyo**: Si la institución u organización beneficiada no cuenta con los recursos para proporcionar apoyo interno, se sugiere tener en cuenta sólo las aplicaciones con soporte profesional. La necesidad de soporte profesional puede ser mejorado por la presencia de una buena comunidad de apoyo.
- **Implementación del lenguaje:** La adopción de software de código abierto a menudo requiere cierto trabajo de personalización y de apoyo interno. Puede ser importante para

[1] *Independent software vendor (ISV):* Persona o empresa que desarrolla software. Esto implica una organización que se especializa en software y no es parte de los sistemas informáticos o el fabricante de hardware. En general, los ISVs crear aplicaciones de software en lugar de software del sistema, tales como sistemas operativos y sistemas de gestión de bases. [6]

elegir las aplicaciones de acuerdo a la experiencia del experto disponible en la empresa o institución.

- **Las revisiones por terceros:** Adoptantes puede investigar las revisiones hechas por terceros en relación con el software para ser considerado.
- **Libros:** La disponibilidad de libros sobre el software es un fuerte indicador del nivel de madurez y adopción.
- **Seguimiento por los analistas del sector, como Gartner o IDC:** Otro indicio de la viabilidad de la adopción del *software* es la disponibilidad de informes de investigación realizados por las principales empresas de investigación de mercado como Gartner e IDC.

C. Añadir más indicadores de viabilidad interna a la lista si aplica: El nuevo paquete de software que se adopte ha de cumplir con varios requisitos fundamentales. Estos requisitos deben ser incluidos en la lista de indicadores de viabilidad para la evaluación rápida.

D. Crear una política de criterios de aprobación: Después de compilar una lista de indicadores de viabilidad para una evaluación rápida, los adoptantes necesitan crear una política de cuales resultados de la evaluación son aceptables. La mayor parte de los criterios de viabilidad que se han seleccionado dan como resultado una respuesta afirmativa o negativa, pero en caso de que la respuesta no sea un rotundo si o no es posible tener una categoría de medición intermedia, esto es análogo a las luces rojas o verdes de un semáforo, con una categoría intermedia amarillo. Una política aceptable podría ser “todos los verdes”, o “amarillos no más de 2”, etc.

E. Evaluar cada componente de software con la lista de indicadores de viabilidad: Después de haber creado una lista de filtros de la viabilidad de aplicación y la política de paso, los evaluadores pueden comenzar a realizar la evaluación rápida de los candidatos de software. El resultado de la evaluación rápida determina si el software candidato debe seguir para el proceso de evaluación total en OpenBRR.

A.2 Fase 2: Evaluación del uso Objetivo:

A. Selección de las Categorías de Evaluación más importantes: El objetivo de esta fase es la clasificación, elección y ponderación de las categorías de evaluación, para con ello obtener las métricas para evaluar el *software* candidato. La métrica se define como una propiedad medible de un proyecto de software libre o de código abierto, FLOSS. Algunos ejemplos de estas son: el número de libros publicados acerca del software, el número de comités del proyecto, y el nivel de las actividades de prueba. Es importante organizar el proceso de evaluación en áreas de interés, o categorías de evaluación. Inicialmente en el método OpenBRR se definen las 12 categorías para la evaluación de software.

Tabla A. 1 Categorías para la evaluación de software.

CATEGORÍA DE EVALUACIÓN	DESCRIPCIÓN
Funcionalidad	¿Cómo el software podrá satisfacer las necesidades del usuario?
Usabilidad	¿Qué tan buena es la interfaz de usuario? ¿Qué tan fácil de usar es el software para los usuarios finales? ¿Qué tan fácil es el software para instalar, configurar, implementar y mantener?
Calidad	¿De qué calidad son el diseño, el código y las pruebas? ¿Qué tan completos y libre de errores son?
Seguridad	¿Qué tan bien el software maneja la seguridad? ¿Qué tan seguro es?
Rendimiento	¿Qué tan bueno es el desempeño del software?
Escalabilidad	¿Qué tan escalable es para un ambiente amplio?
Arquitectura	¿Qué tan buena es la arquitectura de software? Cómo tan modular, portátil, flexible y extensible, abierto y fácil de integrar es?
Compatibilidad	¿Qué tanto es el componente de software compatible?
Documentación	¿De qué calidad es la documentación referente al software?
Adopción	¿Qué tan bien es el componente aprobado por la comunidad, el mercado y la industria?
Comunidad	¿Qué tan activa y dinámica es la comunidad para el software?
Profesionalismo	¿Cuál es el nivel de la profesionalidad del proceso de desarrollo y la organización del proyecto en su conjunto?

Se define una evaluación de software desde un aspecto específico, como un Puntaje de Categoría. Un Puntaje de Categoría se obtiene mediante la agrupación de varias métricas que miden los mismos aspectos. La forma en que la puntuación de una categoría es calculada puede diferir de la forma en que otra categoría es medida, pero los resultados deben utilizar la misma escala (1 a 5). Una métrica puede contribuir a varias categorías de diferentes maneras: por ejemplo, el ciclo de liberación (*release*) de Fedora que se realiza cada seis meses indica un alto nivel de "vida" de la comunidad, pero un bajo nivel de estabilidad.

Al trabajar con OpenBRR, se sugiere no incluir todas las doce categorías de evaluación, según la orientación funcional del software, la relevancia de ciertas categorías de evaluación pueden no ser lo suficientemente significativas para justificar los esfuerzos en la recolección de datos o que la importancia de las categorías de evaluación cruciales pueda ser diluida por categorías de menor trascendencia, se sugiere que se enfoque en no más de siete (7) categorías de evaluación a fin de que sea una ayuda para decidir la contribución en la ponderación de las calificaciones de la categoría de cada área, para lo cual debe seguirse lo siguientes pasos:

- Clasificación de la importancia de las categorías de la evaluación de uno a doce (1-12) con respecto a la orientación funcional del software.
- Definición de un punto de corte en la lista de categorías de evaluación, una vez ordenadas las categorías por orden de relevancia, se elige un punto en que la diferencia de importancia entre una categoría y la siguiente es grande. Las categorías que están por encima del punto de corte deben ser tenidas en cuenta mientras que las que quedan por debajo de línea de

corte se deben descartar e ignorar. La metodología sugiere que los evaluadores se centren en 7 o menos categorías de evaluación.

B. Selección de los factores de ponderación adecuados

Una vez que se ha seleccionado las Categorías de Evaluación más importantes, se procede a determinar la cantidad o niveles de contribución de cada categoría al resultado final. Estos niveles de contribución son los factores de ponderación que son representados en porcentajes, los cuales deben sumar el 100%. Para determinar una buena distribución de los factores de ponderación, en el supuesto de que se tengan 7 categorías de evaluación, la metodología sugiere que los evaluadores sigan las siguientes pautas:

- Ordenar las categorías en base a la importancia obtenidos en el paso anterior.
- Asignar un factor de ponderación a la categoría que tiene el rango medio (mitad de la lista ordenada), en el caso de las 7 categorías, la categoría que ocupa la mitad, el cuarto puesto inicial debe obtener una ponderación del 15%, y en base a esto se pondera las otras categorías.
- Asignar ponderaciones a las categorías restantes. El proceso de asignación de ponderados sigue un "patrón de zigzag" para las categorías. En el caso de las siete categorías, la categoría que ocupó el cuarto lugar tendría un peso inicial de 15%. Después de eso, las categorías más altas en la lista (en el primer puesto al tercero en importancia), se le asignaría ponderaciones de 15% o más, y para las categorías inferiores en la lista (en el quinto puesto al séptimo en importancia) se le asignaría ponderación no superior al 15%. Una muestra de una distribución de peso buena para las 7 categorías pueden ser: la categoría más importante: el 25%, el segundo más importante categoría: 20%, la tercera más importante de la categoría: 15%, mediana de la categoría: 15%, el quinto clasificado de la categoría: 10%, sexto del ranking categoría: 10%, y menos importante categoría de los siete: 5%.
- Debe asegurarse que la suma de los pesos de las categorías a evaluar es de 100%.
- Posibilidad de poder hacer ajustes finales si es necesario.

C. Normalización de medidas de las métricas

Todas las mediciones o métricas dentro de una categoría, ya sean cualitativos o cuantitativos, deben ser comparados con una escala normalizada que permita realizar la medición significativa y tenga sentido la comparación, las medidas cuantitativas, como el número de descargas de un paquete de software, son relativamente fáciles de normalizar, mientras que las métricas cualitativas, que también tienen que ser normalizadas, presenta un grado de dificultad por que el proceso es subjetivo. Para crear un modelo OpenBRR normalizado, es importante normalizar los datos en bruto (*raw data*) de estas métricas.

Se utiliza una escala básica de 1 a 5 en donde los valores se podrían relacionar como se observa en la tabla A.2

Tabla A. 2 Escala normalizada de medidas de las métricas OpenBRR.

VALOR	CONCEPTO
1	Inaceptable
2	Pobre
3	Aceptable
4	Muy Bueno
5	Excelente

Algunas métricas no necesitan usar todos los valores listados en la tabla 6, las métricas de tipo booleana (verdadero o falso), tiene 2 posibles valores, a los cuales se les asignaría el valor 1 si es falso ó el valor 5 si es verdadero. Existen otras formas de cuantificar métricas en la documentación de la metodología en OpenBRR [7].

Cada métrica dentro de cada Categoría debe tener un factor de peso para diferenciar la importancia de la métrica dentro de una categoría particular.

Las métricas de funcionalidad se procesan de una forma distinta a las métricas de las otras categorías. Cada tipo de aplicación de software tiene un conjunto de características que deben ser cumplidas por las aplicaciones que caigan en ese tipo, y además algunas aplicaciones pueden proveer otras características adicionales que también deben ser tenidas en cuenta en el momento de evaluar esta categoría. Se debe analizar que característica posee el sistema y compararlas con un conjunto estándar de características de la familia de ese sistema. Una vez obtenida la lista de características comunes a los sistemas, se siguen los siguientes pasos:

- Asignar una puntuación según la importancia de la característica a todas aquellas que se encuentren en la lista de características comunes, usando una escala de 1 a 3, en donde 1 indica que es una de las menos importantes y 3 indica que es una de las más importantes.
- Comparar la lista de características del sistema contra la lista de estándares, para cada característica de la aplicación encontrada en la lista estándar se debe sumar la puntuación de importancia en una suma acumulativa. Las características estándares que no sean cumplidas por el sistema se restaran de la suma acumulativa.
- Si el sistema tiene características extraordinarias (no se encuentra en la lista de estándares), se debe asignar un grado de importancia a la misma y agregarla a la suma acumulativa.
- Luego de culminado el proceso para todas las características del sistema se debe dividir el total obtenido sobre el máximo total posible de las características estándares. Este índice es la puntuación de las características funcionales. Es posible que sea mayor que 100% o menor que 0%, de esta manera se premia el tener funciones extras y se castiga el no tener aquellos estándares.
- Luego de obtener el porcentaje se debe comparar a la siguiente lista de puntuación para obtener el puntaje de la categoría funcional.

Tabla A. 3 Lista de puntuación para el puntaje de categoría funcional en OpenBRR.

PORCENTAJE OBTENIDO	PUNTUACIÓN	DESCRIPCIÓN
Menos de 65 %	1	Inaceptable
Entre 65 % y 80 %	2	Malo
Entre 80 % y 90 %	3	Aceptable
Entre 90 % y 96 %	4	Muy Bueno
Mayor que 96 %	5	Excelente

A.3 Fase 3: Recolección de datos y procesamiento

Para poder cuantificar las categorías es necesario recoger información, para ello se utiliza diferentes fuentes de consulta: sitios web de los proyectos, forjas, artículos técnicos, libros publicados sobre la aplicación, opiniones, etc.

A.4 Fase 4: Selección del aplicativo

Para obtener los resultados de la comparación de las aplicaciones se utilizan plantillas en hojas electrónicas proporcionadas por los desarrolladores de la Metodología.

A.5 Criterios de las categorías Representativas.

Como se explico en A.2 los criterios para medir la funcionalidad son establecidos por el evaluador, pero para las otras 11 categorías estos se encuentran establecidos en [1] y son los que se pueden ver en la tabla A.4

Tabla A. 4 Criterios de las Categorias.

CATEGORIA	METRICAS	DESCRIPCION	PUNTAJE				
			5 EXCELENTE	4 BUENO	3 ACEPTABLE	2 POBRE	1 INACEPTABLEE
USABILIDAD	Interfaz de usuario: Experiencia de Usuario Final	Este mide qué tan bien la interfaz de usuario es percibida por un usuario final.	Simple e intuitivo, la información está bien organizada, no se requiere manual		Toma poco tiempo para aprender, un poco de información organizada, algún uso del manual.		Información compleja, demasiado o ninguna organización obvia, no se puede utilizar sin un manual.
	Tiempo para configurar los requisitos previos para la instalación de software de código abierto	Tiempo / esfuerzo necesario para establecer un sistema, con todos los requisitos previos satisfechos. Esto no incluye sistema operativo.	< 10 minutos	10 - 30 minutos	30 min - 1 hora	1 - 4 horas	> 4 horas
	Tiempo para instalación y configuración del software.	El tiempo que toma para conseguir la gratificación instantánea, muestra si el proyecto está pensando para conseguir que los usuarios inicien a trabajar tan rápidamente como sea posible.	< 10 minutos	10 - 30 minutos	30 min - 1 hora	1 - 4 horas	> 4 horas

CATEGORIA	METRICAS	DESCRIPCION	PUNTAJE				
			5 EXCELENTE	4 BUENO	3 ACEPTABLE	2 POBRE	1 INACEPTABLEE
CALIDAD	Número de versiones menores en los últimos 12 meses	Esto mide las actualizaciones planificadas y correcciones de errores. Normalmente, paquetes de servicio en productos comerciales.	2		1 o 3		0 o > 3
	Número de comunicado de point/patch en los últimos 12meses	Normalmente, los oficiales de point/patch liberados son correcciones de errores P1 como bloqueos, memoria y vulnerabilidades de seguridad.	3 – 4		1 - 2, o 5 - 6		0 o > 6
	Número de bugs abiertos en los últimos 6 meses	Este mide la calidad de uso del producto.	< 50	50 - 100	100 - 500	500 - 1000	> 1000
	Número de errores corregidos en los últimos 6 meses (en comparación con el # de errores abiertos)	Este mide la rapidez con errores fijos.	> 75%	60% - 75%	45% - 60%	25% - 45%	< 25%
	Número de errores P1/críticos abiertos	Este mide la gravedad de los problemas de calidad encontrados	0	1 - 5	5 - 10	10 - 20	> 20
	La edad promedio de los errores de P1 en los últimos 6 meses	La edad promedio de errores de P1 en los últimos 6 meses.	< 1 semana	1 - 2 semanas	2 - 3 semanas	3 - 4 semanas	> 4 semanas
SEGURIDAD	El número de vulnerabilidades de seguridad en los últimos 6 meses que son desde moderadas a extremadamente críticas	Este mide la calidad relacionada con las vulnerabilidades de seguridad. ¿Cómo es susceptible el software a vulnerabilidades de seguridad?.	0	1 - 2	3 - 4	5 - 6	> 6
	El número de vulnerabilidades de seguridad que siguen abiertas (sin parchear)	Esto mide, si el proyecto es capaz de resolver todos los problemas de seguridad.	0	1	2	3 - 5	> 5

CATEGORIA	METRICAS	DESCRIPCION	PUNTAJE				
			5 EXCELENTE	4 BUENO	3 ACEPTABLE	2 POBRE	1 INACEPTABLEE
	¿Hay una página dedicada a la información (página web, wiki, etc) para la seguridad?	Este mide la conciencia y seriedad con la que el proyecto toma los problemas de seguridad.	Si, en buen estado		Si		No
RENDIMIENTO	Pruebas de rendimiento e informes de referencia disponibles	Estas miden si ha habido alguna prueba de rendimiento realizada y publicada, son puntos de referencia por lo general en comparación con otras soluciones equivalentes.	Si, con Buenos resultados.		Si		No
	Ajuste de rendimiento y configuración	Esto mide si existe alguna documentación o herramienta para ayudar a afinar el componente de rendimiento.	Si, Extensivo.		Si, Algunos.		No
ESCALABILIDAD	Referencia de implementació n	Esto mide si el software es escalable y probado en el uso real a través de una implementación real.	Si, con publicacione s de los usuarios.		Si		No
	Diseñado para la escalabilidad	Este mide si el componente ha sido diseñado con la escalabilidad en mente. ¿Es segura para los subprocesos? ¿Se ejecuta en un entorno de clúster? Puede H / W resolver problemas de rendimiento?	Si, extensivo.		Si, algunos.		No
ARQUITECTURA	¿Hay algun plug-in de terceros?	Esto mide el diseño para la extensibilidad a través de plug-ins de terceros.	> 10	6 - 10	2 - 5	1	0
	API pública / Servicio Exterior	Permite extensiones a través de una API pública, también se muestra el diseño para la personalización.	Si, extensivo		Si		No
SOPORTE	El volumen medio de la lista de correo general en los últimos 6 meses	La lista de correo general, es el primer lugar donde la gente va en busca de ayuda gratuita.	> 720 mensajes por mes	300 - 720 msg por mes	150 - 300 msg por mes	30 - 150 msg por mes	< 30 msg por mes

CATEGORIA	METRICAS	DESCRIPCION	PUNTAJE				
			5 EXCELENTE	4 BUENO	3 ACEPTABLE	2 POBRE	1 INACEPTABLEE
	La calidad del apoyo profesional	El apoyo profesional que ayude a afinar para el despliegue local y la solución de problemas es siempre deseable.	Instalación + solución de problemas + Integración / Apoyo personalizado		Solo Soporte de Instalación.		No hay soporte profesional.
DOCUMENTACION	La existencia de varios tipos de documentación	Una suite de una buena documentación debe incluir la documentación de varios grupos de usuarios en varios formatos.	Instalar / implementar, usuario, administración, optimización, mejora, desarrollo, la documentación está disponible en múltiples formatos (pdf, html sola, de varios archivos html).	Instalar / despliegue, usuario, administrador, guías de actualización disponibles en varios formatos	Instalar / implementar y guía de usuario están disponible	Sólo basado en texto, existe documentación de instalación	Sin la documentación adecuada. Un archivo README no cuenta
	User contribution framework	Los mejores guías a menudo provienen de la entrada /muestras del usuario. Esta es la opinión de personas que han utilizado los productos.	Las personas pueden contribuir, y las contribuciones se editan/filtran por los expertos		A las personas se les permite contribuir		Los usuarios no pueden contribuir.
ADOPCION	¿Cuántos títulos de libros hace Amazon.com registra para la consulta Power Search: "Asunto: equipo y el título: nombre del componente"?	La disponibilidad de libros es, sin duda buena. Una forma estándar de conteo de los libros disponibles es importante.	> 15	6 - 15	3 - 6	1 - 3	0
	Referencia de implementación	Este mide si el software es escalable y probado en el uso real a través de una implementación real.	Si, con publicaciones de los usuarios.		Si		No
COMUNIDAD	El volumen medio de la lista de correo general en los últimos 6 meses	La lista de correo general es el lugar donde la comunidad se ayuda.	> 720 mensajes per mes	300 - 720 msg por mes	150 - 300 msg por mes	30 - 150 msg por mes	< 30 msg por mes

CATEGORIA	METRICAS	DESCRIPCION	PUNTAJE				
			5 EXCELENTE	4 BUENO	3 ACEPTABLE	2 POBRE	1 INACEPTABLEE
	Número de contribuyentes de código en los últimos 6 meses	Los contribuyentes de código por lo general promueven la construcción de la comunidad en torno al proyecto. Cuanto mayor sea el número de contribuyentes de código, mejor será el apoyo de la comunidad.	> 50	20 – 50	10 - 20	5 - 10	< 5
PROFESIONALISMO	Controlador del Proyecto	El conductor del proyecto lleva a cabo la gestión de los recursos (dinero), recogida, etc	Fundación independiente con el apoyo de las corporaciones(estilo Apache / OSDL)	Corporación (estilo de MySQL)		Grupos	Individuales
	Dificultad para entrar en el equipo principal de desarrolladores	Para asegurar la calidad del software, los proyectos maduros deben ser selectivos en la aceptación de desarrolladores. Los nuevos proyectos a menudo no tienen otra opción.	Sólo después de haber sido "comitter" externo por un tiempo		Más bien difícil, debe contribuir con parches aceptados por algún tiempo		Cualquiera puede entrar

BIBLIOGRAFÍA

[1] A. I. Wasserman, M. Pal, y C. Chan, "Business Readiness Rating Project", "BRR Whitepaper 2005 RFC 1". [En línea] Disponible en: http://pascal.case.unibz.it/retrieve/1095/BRR_whitepaper_2005RFC1.pdf [Accedido: Abr. 12, 2011]

[2] A. Das y A. I. Wasserman, "Using FLOSSmole Data in Determining Business Readiness Ratings", *Workshop on Public Data about Software Development - WoPDaSD 2007*, 2007, Limerick, Irlanda.

[3] W. Leister; N. Christophersen, "INF5780 Compendium Autumn 2011: Open Source, Open Collaboration and Innovation", *publications.nr.no*, Sept. 2011. pp. 76-79. [En línea]. Disponible en: http://publications.nr.no/Compendium-INF5780H11.pdf. [Accedido: Ene. 28, 2012].

[4] Open Source Initiative, "The Open Source Definition (Annotated)", *opensource.org*. [En línea]. Disponible en: http://www.opensource.org/osd.html. [Accedido: Abr. 30, 2011].

[5] Free Software Foundation, "¿Qué es el copyleft?,". [En línea]. Disponible en: http://www.gnu.org/copyleft/copyleft.es.html. [Accedido: Ene. 20, 2012].

[7] A. I. Wasserman, M. Pal, y C. Chan, "The Business Readiness Rating Model: an Evaluation Framework for Open Source", *Proc. Of Workshop on Evaluation Techniques for Open Source Software, 2nd Int'l. Conf. on Open Source Systems*, 2006, Como, Italia.