

Metrics of software reuse for free and Open Source software

Project	ODETTE
Author	RAMEL Sophie Centre de Recherche Public Henri Tudor 29, Avenue John F.Kennedy L-1855 Luxembourg-Kirchberg Tél.: +352 42 59 91 - 1 Fax: +352 42 59 91 – 777 http://www.tudor.lu/
Creation Date	25/03/2005
Version and Date	« 1.0 », 15/07/2005

Table of Contents

1Introduction.....	1
2Reuse of libraries/components.....	3
3Frameworks reuse.....	3
4Metrics of Amount of Reuse.....	4
4.1Existing Amount of Reuse Metrics.....	4
4.2Granularity of Metrics.....	4
4.3Application of this Measure to Java Programs.....	5
4.4Measure for C/C++ Programs.....	7
5Comparing with reuse in proprietary software.....	9
6Conclusion.....	9
7References.....	10
8Appendix – lists of projects used for measures.....	11
8.1Program to calculate the ERL of Java applications based on the JDepend results.....	11
8.2ERL applied on Java stable projects randomly selected on sourceforge.net.....	14
8.3ERL applied on Famous Open Source Java projects from java-net.....	15
8.4Script to apply the ERL measure to a RPM package.....	16
8.5ERL applied on C/C++ projects on a SuSE Linux 9.2 distribution.....	18

1 INTRODUCTION

The results of studies on software reuse made a few years ago usually showed that software reuse could be improved, and that it would improve the quality and the productivity of software [4]. Because there is more and more software developed using a Free software development methodology, and because a lot of Free software now comes from many generations of reused software, we will study the quantity of reuse in Free software.

This document will thus concentrate on the measurement of the quantity of reuse, and not on organizational or financial consequences of reuse. This choice also corresponds well to the domain of Free software, since organizational and financial aspects are very diverse and often missing in Free software.

Free software is by nature well suited to reuse, as well as to the measurement of reuse: allowing software reuse is one of the main objectives of free software, since the goals of free software, as defined by Richard Stallman ([1]), are to allow the freedom of use, of modification, of redistribution and of redistribution of modified versions of software. Three of these four main objectives of free software are thus directly linked to software reuse.

This is also the case with Open Source software, because Open Source allows the same "freedoms" than Free software: Open Source is very similar to Free, except that there are more licenses, approved by the Open Source Initiative (<http://www.opensource.org/>), which propose the same concepts as Free software, except for the notion of "copyleft" sometime present in Free software. This notion forces modified versions of these programs to be released under a Free license.

The main consequences of reuse of Open Source software by comparison to reuse of Free software are not on the program itself but on the programs reusing it, since software reuse can then be limited to only one level: for Open Source software, the program is reused but it is possible that the derived program won't be reusable since it can be distributed under a non Open Source or Free license, while for Free software the level of reuse is unlimited (all the programs produced by reuse will be reusable since they will still be Free software).

The rest of the document concerns both these notions of Free and Open Source Software, often referred to as "FOSS".

Measurement of software reuse is made easier for Free software than for proprietary software since the code can be directly inspected to measure reuse. In addition, because license and copyright headers are usually well respected in Free software, the origins of reused software can be easily found; For example, the GPL license information page of the GNU project [2] informs the users of how to modify these headers when they reuse Free software:

If you have copied code from other programs covered by the same license, copy their copyright notices too. Put all the copyright notices together, right near the top of each file.

As explained in the document "Software Reuse in Free software: state of the art" ([2]), software reuse on a technical level can be divided into conceptual reuse (reuse of models, concepts or ideas) and program reuse (reuse of components, frameworks and code). Conceptual reuse being very difficult to measure, we will concentrate on reuse of programs.

2 REUSE OF LIBRARIES/COMPONENTS

The goal of this kind of reuse is to use a component in a program, in order to use one or more of the functionalities offered by the program. More details on components reuse can be found in [6].

In order to measure the amount of components reuse, we need to differentiate direct and indirect reuse:

- direct: the component is used directly in the program
- indirect: the component is needed by another component which is used in the program. This is a kind of "transitive" reuse.

Depending on the packaging mode, it can be easier or harder to find the number of components used directly or indirectly: for programs including all the needed libraries one has to read the documentation or look at the code to see which libraries are used directly and which are used indirectly. On the contrary, for programs packaged with dependencies clearly defined, it is easier to retrace the libraries needed directly and the others.

Limit between system use and reuse

A lot of programming languages provide basic functionalities by default with the language. We cannot consider as reuse these modules since they are used by everybody and can be considered as part of the language. As a consequence, for our study, it is difficult to differentiate which of these components are default components provided by programming languages, and which are components that have to be counted as reuse.

Two possibilities are:

- only consider functions/classes that need to be installed in addition to the language (example: installation of a SOAP server, axis, but not the XML libraries included by default in Java). This poses problem for languages which are not always distributed with the same libraries, like C on Linux/Unix systems
- consider reuse only when the component provides a certain level of complexity and specificity (example: consider JDBC calls or manipulations of XML like reuse, but not writing into a file)

The first solution is less subjective and thus simpler to measure, but it highly depends on the programming language or operating system providing the default components, and it also doesn't take into account some components as soon as they have been integrated into default distributions: some components which are not installed by default with some versions of the language or operating system become packaged by default on following versions (for example this is the case with XML "DOM" and "SAX" libraries in Java). The second solution is perhaps more realistic, because it only considers as reuse the usage of sufficiently complex components. However, to decide which components are sufficiently complex is a highly subjective decision. As a consequence, in order to stay objective and to get comparable results, we will only consider the first solution.

3 FRAMEWORKS REUSE

As explained in "Software Reuse in Free software: State of the art" [6], framework reuse is quite similar to components reuse, except that all the components of the framework are reused in a consistent way. Thus, we will not differentiate the measure of framework reuse from the the measure of components reuse.

4 METRICS OF AMOUNT OF REUSE

These are the primary metrics we will focus on; they do not measure the consequences of software reuse, but simply how much code does a program reuse.

Initially, code reuse was often described as only concerning modules of codes that were included (and potentially modified) in the code of the system. For Object Oriented languages, this summed up to verbatim reuse and inheritance. However, this does not take into account components-based reuse, which is defined by the use of an external component in order to achieve a goal needed by the system, whether it is included in the code of the system or not. FOSS products provide facilities to reuse code in both ways, by allowing to copy code from a FOSS product into its own program and modify it, or by allowing to use other FOSS products as components. It is this second kind of reuse, components-based reused, which also brings a better maintainability of the application, that is more significant of FOSS products, and that we will thus consider and measure.

In the context of reuse by inclusion, object oriented languages are considered a specific case because of the ability of reuse by inheritance. However, since we will concentrate on components-based reuse, this specificity is not relevant. Moreover, what is relevant is the facility with which a language allows the programmer to use external components. Thus, the measure of reuse will be made for different well-used programming languages.

4.1 Existing Amount of Reuse Metrics

These kinds of metrics usually work on "items", which are defined according to the granularity chosen (see next chapter).

Reuse Level

This metric, defined by Frakes and Terry ([5]), calculates the number of items reused in the program related to the total number of items. It is one of the simplest and well know reuse metric.

Reuse Frequency

This metric, defined by Frakes and Terry ([5]), calculates the number of references to reused items related to the total number of references.

It has been shown to be highly correlated to the Reuse Level metric [7].

Reuse Density

This metric, defined by Benedicenti, Succi, and Vernazza ([8]), measures the number of reused item related to the total number of instructions.

We also differentiate the "Internal" and "External" variants of these metrics: the internal metrics measure the reuse of items that are part of the program, while the external metrics measure the reuse of items that are external to the program.

4.2 Granularity of Metrics

The amount of reuse can be measured by comparing the number of reused "items" with the total number of "items" [5], where items depend on the granularity chosen (for example function/method, class, package, file, program etc). As a consequence, the application of amount of reuse metrics requires to select the granularity on which to apply them.

The granularity chosen can have consequences on the relevance of the measures.

Lines of code

We can use the number of lines of a program. In this case, to measure the reuse level, we have to calculate:

$$\frac{\text{number of reused lines}}{\text{total number of lines}}$$

This in fact measures only what we call the "code reuse" (direct reuse of code from another program), because it considers that software reused is present directly in the form of lines of codes.

However, in the case of Free software, this is not the main kind of reuse: reuse of code is not very significant compared to reuse of components or frameworks, probably because it requires to know and understand the code to reuse, and it is difficult for the programmer to make any update of the reused code. In addition, it is usually only used for very small amounts of code. As a consequence, we will choose another granularity for this study.

Functions / Methods

This granularity allows to count the number of reused functions or methods, which often comes from external components. It is thus well adapted to measure components software reuse, even if not the simplest metric to implement.

Packages

In programming languages containing the notion of packages, this notion helps developers to structure their code in different ways, often linked with a set of functionalities (for example a package containing some specific business functionalities) or of specificities to some applications of the product (for example a package containing the user interface for one platform). In addition, there are tools to measure packages dependencies. This granularity is thus well suited to measure the reuse of external components, and also relatively easy to implement.

In order to apply this granularity to the "External Reuse Level" metric, we have to divide the total number of external packages used by the number of packages in the program (including the external packages):

$$\text{External Reuse Level} = \frac{\text{number of external packages used}}{\text{number of internal package} + \text{number of external packages used}}$$

Also note that it is important to apply this measure recursively on packages: in order to be complete, we not only have to count the external packages used in the program, but also the packages used by these external packages (whether they are in the same component or in another component). Otherwise we will not count the total number of packages used by the program when it runs.

4.3 Application of this Measure to Java Programs

One of the strengths of the Java programming language is the number of components available to reuse, and the language mechanisms which facilitate reuse by providing external package formats (Java archives ".jar" files) as well as loading facilities (Java "Class Loader" mechanism). Because of this, Java programs usually reuse more components than programs in other programming languages.

The application "JDepend" (<http://www.clarkware.com/software/JDepend.html>) provides quality metrics for Java programs based on dependencies. Because it begins by calculating package dependencies, it can be used to apply our measure: we begin by having JDepend calculate all the dependencies of the program and of all its needed libraries. The result is written in an XML file which can then be interpreted by a small program that we have written, in order to count all

the dependencies coming (directly or indirectly) from the packages which are part of the program.

The conditions to apply this measure are that the ".jar" files of the program and of all its libraries are available, that the code of the program and of the libraries is structured into packages (which is the case most of the times), and also that it is not part of a framework (otherwise the number of reused libraries is immediately very high. Another solution for this is to not take into account the framework dependencies).

Limit between system use and reuse

In order to take into account only "real" reuse, and not simple usages of the language constructs, we have decided to count the packages used only when they were not part of the default installation. To do so, we removed all the packages that were provided with the version 1.4.2 of the Java Development Kit, by considering that at the time of the measure, the JDK 1.4.2 was the most used. We also removed the packages under "javax.servlet", even if they are not part of the JDK, because they are usually provided with servlet engines.

Test cases for this measure

We have applied this measure on different groups of projects which met these conditions:

Randomly selected stable projects

The first group were Open Source projects taken randomly from a project repository (sourceforge.net), after applying the following search criteria: programs in the Java programming language, in the state "stable", which provide a ".jar" file (or for which a .jar is easily generated) and .jar files for the libraries, which is structured in packages, and which is not part of a framework. To select the projects, we used the search tool provided by sourceforge.net, and we selected a project every 50 projects on the result page sorted by "Unix project name", this number varying in case the project found did not match the criteria.

Well known / interesting projects

Another idea was to select projects which are particularly well-known in the Open Source domain.

To do so, we have decided to use the projects listed on the <http://java-source.net/> web site, since they are Open Source Java projects selected because of their interest or fame.

We decided to take into account each first project of a category, which are the projects presented on the home page of the site.

Measures results and interpretation

Results for the randomly selected projects

54 projects of the 58 selected met all the constraints. For these projects, the result has an average of 0.45.

The details of the results can be seen on the appendix ERL applied on Java stable projects randomly selected on sourceforge.net on page 14.

Results for the well known / interesting projects

58 projects were considered for this measure. The result has an average of 0.52. The details of the results can be seen on the appendix ERL applied on Famous Open Source Java projects from java-net on page 15.

Interpretation

An average external reuse level of 50% means that half of the program is done by the external packages. With results of 45%, or even 52% for selected projects, software reuse in these products is thus relatively high, even more if we take into account that this only measures the external components reuse, not the internal components reuse and not the code reuse. In addition, no particular efforts were made on these projects to reuse components or to design them as reusable.

The results for the well known projects are higher than the results for random projects, though not very significantly, which is logical since random projects taken on a web site like sourceforge.net also contain very small projects, often written by only one person and with less efforts on the design.

Something that can be noticed for the two test cases results is that these averages are lowered by a lot of products which don't reuse anything at all. After a closure inspection, it appears that the products having no dependency at all are only the smaller products. An interpretation of this is that these products are in fact not standalone products but products providing a very limited number of functionalities and usually used as libraries in other products.

As a consequence, it is interesting to apply this metric on projects whose ".jar" files (with only ".class" files, and no included dependencies) are bigger than 700 kilobytes. This gives an average of 0.64 for the randomly selected projects (but only 9 of these projects are bigger than 700K), and of 0.57 for the java-source projects (for which 31 projects are bigger than 700K).

4.4 Measure for C/C++ Programs

C and C++ programs don't contain directly the notion of packages as with Java programs, but an equivalent notion called "namespaces". However, they are not used in every C program, and are also more difficult to measure. As a consequence, we will not use this granularity to apply our metric.

C programs use shared libraries, usually named "lib*.so.x" where "*" is the name of the library and "x" the version number. These packages are archives containing different header files that are loaded by programs at runtime. We will thus use them as our measurement unit for software reuse.

Limit between system use and reuse

Since every C or C++ library can as well be considered a part of the system as an external library, it is very difficult to differentiate libraries reused from standard libraries used in a program: one has to decide of a limit, based on the common usage of libraries.

To do so, we have chosen to use the "Linux Standard Base" (LSB) project, which aims at defining a set of standards among Linux distributions. We have chosen to consider the standard libraries proposed in this project as system libraries, and all the others as external libraries reused.

These libraries are:

- From the LSB Core Specification: libgcc_s, libc, libcrypt, libdl, libm, libncurses, libpam, libpthread, libutil, libz
- From the LSB C++ generic Specification: libstdc++
- From the LSB Graphics Specification: libX11, libXext, libSM, libICE, libXt, libGL

There are also other libraries which are part of Linux systems and should not be considered: linux-gate, and ld-linux.

Test case for this measure

A very representative base of Free and Open Source C or C++ programs is the Linux system. Moreover, these programs are easily accessible on Linux distributions, and are usually packaged in a way allowing us to retrieve dependencies information (using package formats like RPM) . As a consequence, we have chosen to apply our measure on a Linux distribution (SuSE 9.2) where programs are packaged using the " RPM " program. We removed the packages that were not considered as Open Source before applying the measure.

Measure

The measure is done with the help of a script (written in Perl), which counts the library dependencies (files with ".so. ") of a RPM file that are not provided by this same RPM, and that are not in the list of standard libraries. It then compares it to the total number of libraries (number of libraries in the RPM file plus this number of external libraries) to calculate the external reuse level.

In addition, because we cannot differentiate C programs from other programs with the RPM header information, we suppose that C programs always use at least one library, usually a standard library (for example libc.so is used in every C program). In case the RPM file doesn't contain any dependency even within the standard libraries list, we consider it is not a C or C++ program and remove it from our list.

In addition, because we use shared libraries to apply this measure, the program itself has to be packaged as a shared library, otherwise we will get non relevant results; we thus also remove from our results the packages which don't contain any ".so" file.

Results

396 RPM packages of the SuSE 9.2 distribution met all these requirements. On these packages, the average of External Reuse Level for libraries (.so files) is of 0.48. Even if this measure relies on libraries in .so files and not on packages like for the Java measures, we can notice that the results of the measures for Java products and for C/C++ products are similar.

This table summarizes the results of the different metrics applied in this study:

Programmin g Language	Granularity for Metric	Origin of projects	Number of projects	Average External Reuse Level
<i>Java</i>	<i>packages</i>	<i>randomly selected on sourceforge.net</i>	<i>54</i>	<i>0.45</i>
<i>Java</i>	<i>packages</i>	<i>projects from www.java.net</i>	<i>58</i>	<i>0.52</i>
<i>C/C++</i>	<i>shared libraries</i>	<i>SuSE 9.2 packages</i>	<i>396</i>	<i>0.48</i>

5COMPARING WITH REUSE IN PROPRIETARY SOFTWARE

Ideally, we would be able to compare the amount of reuse in Free/Open Source software or in software using Free/Open Source software with the amount of reuse of proprietary software not using any Free/Open Source software. In this way, we could show the impact of FOSS on software reuse. Unfortunately, proprietary software are by definition mostly not available, and in addition more and more of them use Free/Open Source software themselves.

As a consequence, we could only try to compare these measures to equivalent measures that could have been made in companies reusing only components from an internal database, also taking into account the fact that Free/Open Source software usually don't make any explicit effort for software reuse.

6CONCLUSION

In this paper we have shown how to measure the amount of reuse in Free or Open Source products, specifically for components reuse, and have applied specific measures to a big number of these products in order to access to relevant statistics. We have shown that the measures of the components reuse are at an average of approximately 50% for Free or Open Source programs made with the C/C++ or Java programming languages.

These results seem very high for products which weren't built with reuse in mind, but because it is difficult to apply the same measure to closed source products, we unfortunately cannot compare these results to proprietary products results. In addition, even if we had access to closed source products, a lot of them reuse Open Source components, and it would still be difficult to measure the impact of Open Source on software reuse. Thus, a direct comparison cannot be made.

However, a 50% reuse level means that half of the programs come from reused components: even without comparisons, these results tend to show that FOSS seems to improve software reuse in quite efficient ways, without all the additional efforts made by companies that have a complete internal reuse policy, with sometimes employees spending time to create reusable components. Since one of the main goals of FOSS is to promote the reuse of products, we can conclude that this goal is indeed achieved, and that this main concept in the Free and Open Source ideas is well respected and implemented in a majority of FOSS products. As software reuse has been shown to improve the quality of software as well as the productivity of software development ([4]), we can also conclude that free and open source software can already be an interesting choice from an economic point of view, even before taking into account the different business models specific to FOSS.

7 REFERENCES

1. Richard Stallman, " Open Sources : Voices from the Open Source Revolution ", O'Reilly 1999
2. "How to use the GPL or LGPL", <http://www.gnu.org/licenses/gpl-howto.html>
3. Rapport "La notion de réutilisation logicielle et sa mise en place dans une entreprise ", Daniel Erésué, 10/2001
4. " Effects of Reuse on Quality, Productivity, and Economics ", [Wayne C. Lim, IEEE Software Septembre 1994](#)
5. William Frakes, Carol Terry, " Software Reuse and Reusability Metrics and Models ", 1995
6. "Software Reuse in Free software: state of the art", Sophie Ramel, 2005
7. W Curry, G Succi, M Smith, E Liu, R Wong, 1999, Empirical Analysis of the Correlation between Amount-of-Reuse Metrics in the C Programming Language
8. Benedicenti, L., G. Succi, T. Vernazza, 1997, Guidelines to Determine the Impact of Code Reuse on Productivity

8 APPENDIX – LISTS OF PROJECTS USED FOR MEASURES**8.1 Program to calculate the ERL of Java applications based on the JDepend results**

```

import org.w3c.dom.*;
import javax.xml.parsers.*;
import java.io.File;
import java.io.FileWriter;
import java.util.Arrays;
import java.util.Vector;
import java.util.Iterator;
/**
Like JDependsResultRecursiveAnalyzer but recursive: JDepends has been called on all the jars needed
by the software
-> the XML file contains all the dependencies
**/
public class JDependsResultRecursiveAnalyzer{
    public static void main(String argv[]){
        if(argv.length>=2){
            JDependsResultRecursiveAnalyzer analyzer = new JDependsResultRecursiveAnalyzer(argv
[0],argv[1]);
            if(argv.length>=4)
                analyzer.showDependencies(argv[2], argv[3]);
            else if(argv.length>=3)
                analyzer.showDependencies(argv[2], null);
            else
                analyzer.showDependencies("", null);
        }
        else
            System.out.println("JDependsResultRecursiveAnalyzer projectName xmlFile
rootPackageName");
    }

    private Document doc;
    private String _projectName;
    public JDependsResultRecursiveAnalyzer(String projectName, String filename){
        this._projectName=projectName;
        doc=this.readDocument(filename);
    }

    /* print the package dependencies for the application */
    public void showDependencies(String rootPackageName, String rootPackageName2){

        System.out.println("showDependencies, rootPackageName="+rootPackageName+",
rootPackageName2="+rootPackageName2);

        //depsResult contains the name of external packages needed
        Vector depsResult=new Vector();
        //nbPackages contains the number of internal packages of the program
        int nbPackages=0;

        if(doc!=null){
            Element jdependEl=(Element) doc.getDocumentElement();
            if(jdependEl!=null){
                NodeList packagesList=jdependEl.getElementsByTagName("Packages");
                if(packagesList.getLength()>0){
                    Element packagesEl=(Element) packagesList.item(0);

                    NodeList packageList=jdependEl.getElementsByTagName("Package");
                    int nb_packages=packageList.getLength();
                    for(int i=0;i<nb_packages;i++){
                        Element packageEl=(Element) packageList.item(i);
                        String name=packageEl.getAttribute("name");

```



```

Element rightPackageEl=null;
if(doc!=null){
    Element jdependEl=(Element) doc.getDocumentElement();
    if(jdependEl!=null){
        NodeList packagesList=jdependEl.getElementsByTagName("Packages");
        if(packagesList.getLength()>0){
            Element packagesEl=(Element) packagesList.item(0);

            NodeList packageList=jdependEl.getElementsByTagName("Package");
            int nb_packages=packageList.getLength();
            int i=0;
            while(i<nb_packages && rightPackageEl==null){
                Element packageEl=(Element) packageList.item(i);
                String name=packageEl.getAttribute("name");
                if(name.equals(pkgName)) //right package
                    rightPackageEl=packageEl;
                i++;
            }
        }
        else
            System.out.println("No Packages element");
    }
    else
        System.out.println("No root element");
}
else
    System.out.println("Document is null");

return rightPackageEl;
}

/**
test if a package is a 'system' package: a package given in the JRE 1.4.2
**/
private boolean isSystemPackage(String pkgName){
    return (Arrays.binarySearch(JavaSystemPackages.getSystemPackages(), pkgName)>-1);
}
}

```

8.2ERL applied on Java stable projects randomly selected on sourceforge.net

Product	Root Package	ERL	HanziHelper	net.coljac.zhongwen	0.58
A2j	org.jzkit.a2j	0.2	Httpunit	com.meterware.httpunit	0.9
Armi	com.navtools	0.38	instantj	instantj	0.29
BIE	com.webdeninteractive	0.81	j2xtreme-xbean	com.j2xtreme.xbean	0.92
ChessY	dav.chess	0	Jaminid	com.prolixtech.jaminid	0.33
Conges	fr.sgauchet.conges	0.95	JarBuilder	ch.fulgur.jarbuilder	0
Dejavu	net.sf.dejavu	0.4	Javadict-client	mt.rcasha.dict.client	0.5
Dozer	net.sf.dozer	0.95	Jaxor	net.sourceforge.jaxor	0.82
Edenlib	com.anthoniyeden	0.79	Jconfig	org.jconfig	0.23
Esw	net.sourceforge.esw	0.39	Jep	org.nfunk.jep	0
fmp	fmp	0.82	Jgrinder	com.objectwave	0.42
Galet	galet	0.63	Jmicr	com.sourceforge.jmicr	0.25
Gfa	org.bitmetrics.gfa	0			

Jox	com.wutka.jox	0.95	Scdoc	com.implementorsolutions	0
Jrex	com.karneim.util	0	Simkin	simkin	0.5
JsynthLib	org.jsynthlib.core	0.44	Srcgen	net.eejits.srcgen	0.5
Jvtelnet	cz.dhl.term	0.5	swiesnerPDFCreator	de.stephanwiesner.pdfcreator	0.99
Kiga3000		0.04	Thinlet	thinlet	0
Lipidia	org._3pq.lipidia	0.76	triveni	triveni	0.33
Majix	com.tetrasix.majix.	0.67	VASSAL	org.vassalengine	0.59
Metagraf		0	Wapide	wapide	0.75
Mt503	org.mozillatranslator	0.44	wicket	wicket	0.42
Neoview	org.neoview	0.25	Wrapper	org.tanukisoftwrapp er.com.silveregg.wrapper	0.2
Ojalgo	org.ojalgo	0.76	Xflow	xflow	0.9
Openmodeling	org.openmodeling	0.99	Xml-writer	com.megginson.sax	0
Owxv3	com.raptus.owxv3	0.58	Zen-garden	net.sourceforge.zengarden	0.33
Photomgr	photomgr	0.32		Average	0.46
Poolit	org.ascentphase.poolit	0			
Quartz	org.quartz	0.55			
RetsTransaction	com.mris.rets	0.45			

8.3 ERL applied on Famous Open Source Java projects from java-net

Product	Root Package	ERL			
ant	org/apache/tools/ant	0.42	ganttproject	net/sourceforge/gant tproject	0.79
antlr	antlr	0	groovy	org/codehaus/groovy	0.86
argoUML	org/argouml	0.47	heritrix	org/archive	0.78
aspectJ	org/aspectj	0.04	hibernate	org/hibernate	0.83
axis	org/apache/axis	0.55	hsqldb	org/hsqldb	0
columba	org/columba	0.35	htmlunit	com/gargoylesoftware/htmlunit	0.92
compierre	org/compiere	0.79	infoglue	org/infoglue	0.64
cvslib	org/netbeans/lib/cvsclient	0	informa	de/nava/informa	0.88
commons-dbc	org/apache/commons/dbcp	0.82	iText	com/lowagie	0.04
displaytag	org/displaytag	0.76	ivata	com/ivata/groupware	0.75
drools	org/drools	0.76	groupware		
egothor	org/egothor	0.5	IZPack	com/izforge/izpack	0.69
exo	org/exoplatform/	0.76	jalopy	de/hunsicker/jalopy/	0.77
findbugs	edu/umd/cs/findbugs	0.58	Javassist	javassist	0
freeCS	freecs	0.15	Javolution	javolution	0
			JBoss	org/jboss	0.64
			JFreeChart	org/jfree	0.04

<i>j-ftp</i>	<i>net/sf/jftp</i>	0.84	<i>scarab</i>	<i>org/tigris/scarab</i>	0.91
<i>JSPWiki</i>	<i>com/ecyrd/jspwiki</i>	0.68	<i>shiftone</i>	<i>org/shiftone</i>	0.21
<i>JUnit</i>	<i>junit</i>	0	<i>spring</i>	<i>org/springframework/</i>	0.8
<i>log4j</i>	<i>org/apache/log4j</i>	0.33	<i>framework</i>		
<i>luxor</i>	<i>luxor</i>	0.64	<i>squirrel SQL</i>	<i>net/sourceforge/squirrel_sql</i>	0.18
<i>MVNForum</i>	<i>com/mvnforum</i>	0.87	<i>struts</i>	<i>org/apache/struts</i>	0.55
<i>nekohtml</i>	<i>org/cyberneko/html</i>	0.92	<i>jakarta</i>	<i>org/apache/catalina</i>	
<i>openJMS</i>	<i>org/exolab/jms</i>	0.67	<i>tomcat</i>	<i>org/apache/tomcat</i>	0.34
<i>oscache</i>	<i>com/opensymphony/oscache</i>	0.74	<i>Trove</i>	<i>gnu/trrove</i>	0
<i>picocontainer</i>	<i>org/picocontainer</i>	0	<i>twister</i>	<i>org/smcp/twister</i>	0.87
<i>proguard</i>	<i>proguard</i>	0.13	<i>velocity</i>	<i>org/apache/velocity</i>	0.62
<i>quartz</i>	<i>org/quartz</i>	0.58	<i>webmail</i>	<i>net/wastl/webmail</i>	0.78
<i>QuickServer</i>	<i>org/quickserver</i>	0.65	<i>xerces</i>	<i>org/apache/xerces/</i>	0.17
<i>quilt</i>	<i>org/quilt</i>	0.42	<i>Xmojo</i>	<i>javax/management</i>	0.6
<i>roller</i>	<i>org/roller</i>	0.87		Average	0.52

8.4 Script to apply the ERL measure to a RPM package

```
#!/usr/bin/perl
#count the external reuse level of a RPM, based on the numbers of external library files (.so) compared to the internal
#date: May - June 2005
#author: Sophie Ramel, CRP Henri Tudor

# standard LSB libraries: to be removed from statistics of reused libs
@stdlibs=
("libgcc_s", "libc", "libcrypt", "libdl", "libm", "libcurses", "libpam", "libpthread", "libutil", "libz", "libstdcxx", "libX11", "libXext", "libSM", "libICE", "libXt", "libGL", "ld-linux", "linux-gate");

# get all the files from the RPM file
$filelist="files.txt";
unlink($filelist);
system("rpm -qlp $ARGV[0] > $filelist");

#transform filelist to keep only libraries, and only once when links
# in order to count the number of internal libraries
open(OFL, "<$filelist");
%new_filelist=();
while(<OFL>){
    $file=$_;
    if($file =~ m/.*\.so\?.*/){
        #check if there is not already the same lib with or without version number
        ($prefix)=(.*)\.so\?.*/;
        if(!exists $new_filelist{$prefix}){ # if value is not empty
            $new_filelist{"$prefix"}=$file;
        }
    }
}
close(OFL);

#count number of internal libraries
$internal= keys %new_filelist;
print("    Number of internal libraries: $internal\n");
```

```

#contains all external dependencies (as keys).
%deps=();

# for each file in RPM, test dependencies
open(FL, "<$filelist");
while(<FL>){
    $file=$_;
    chomp($file);

    # get list of dependencies
    $filedeps="deps.txt";
    system("ldd $file > $filedeps");

    # read dependencies
    open(FD, "<$filedeps");
    while(<FD>){
        if(/^\s*(\S*)s*=>.*){
            $filedep=$+;
            #if it is a library
            if($filedep =~ m/.*\.\so\.*){

                #check if this file is present in the RPM
                $result=system("grep -q \"\$filedep\$\" $filelist");
                #check result + that it is a library
                if($result){
                    if(not(exists($deps{$filedep})){
                        $deps{$filedep}=1;
                    }
                }
            }
        }
    }
    close(FD);
}
close(FL);

# test if there is at least one library. Otherwise: not a C program
@keys=keys(%deps);
$nb=$#keys+1;
if ($nb==0){
    print("          No library -> not a C program !\n");
    exit(1);
}

#remove standard libs
foreach $stdlib (@stdlibs) {
    while (($dep, $value) = each(%deps)) {
        if($dep =~ m/($stdlib)\.\so\.*){
            # remove lib from list
            delete($deps{$dep});
        }
    }
}

#number of dependencies
@keys=keys(%deps);
$nb=$#keys+1;
print("    Number of external libraries: $nb\n");

# External Reuse Level
$erl=$nb / ($nb + $internal);
print("    External Reuse Level: $erl\n");

#write number of dependencies in a file
open(RESULT, ">>results.txt");

```

```
print RESULT "$ARGV[0] $internal $nb $erl\n";
close(RESULT);
```

8.5 ERL applied on C/C++ projects on a SuSE Linux 9.2 distribution

Package tested	int libs	ext libs	ERL	fontconfig- 2.2.96.20040728- 9.i586.rpm	1	2	0.67
id3lib-3.8.3-89.i586.rpm	1	1	0.5	libmikmod-3.1.10- 663.i586.rpm	1	0	0
gnome-keyring-0.2.1- 5.i586.rpm	1	19	0.95	libdvdnav-0.1.10- 2.i586.rpm	1	0	0
kaffeine-0.4.3b- 8.i586.rpm	1	27	0.96	libdc1394-0.9.3- 22.i586.rpm	1	1	0.5
gnokii-0.6.3-3.i586.rpm	1	2	0.67	libkexif-0.0.cvs20040908- 3.i586.rpm	1	19	0.95
liby2util-2.10.6-2.i586.rpm	1	3	0.75	libdv-0.103-2.i586.rpm	1	12	0.92
gtk-qt-engine-0.5- 9.i586.rpm	1	22	0.96	zlib-devel-1.2.1- 74.i586.rpm	1	0	0
mkisofs-2.01-2.i586.rpm	1	1	0.5	curl-7.12.0-2.i586.rpm	1	2	0.67
mad-0.15.1b-30.i586.rpm	1	0	0	fam-2.6.10-122.i586.rpm	1	1	0.5
libmusicbrainz-2.1.1- 3.i586.rpm	1	1	0.5	yast2-bootloader-2.10.17- 2.i586.rpm	1	2	0.67
linc-1.0.3-180.i586.rpm	1	5	0.83	libmng-1.0.8-3.i586.rpm	1	2	0.67
libtunepimp-0.3.0- 3.i586.rpm	1	8	0.89	startup-notification-0.5- 320.i586.rpm	1	0	0
liblcms-1.12-57.i586.rpm	1	2	0.67	xaw3d-1.5E-222.i586.rpm	1	3	0.75
kio_slp-0.4-37.i586.rpm	1	25	0.96	zvbi-0.2.8-2.i586.rpm	1	0	0
fribidi-0.10.4- 485.i586.rpm	1	0	0	pwlib-plugins-v4l-1.8.0- 3.i586.rpm	1	1	0.5
bzip2-1.0.2-347.i586.rpm	1	0	0	lirc-0.7.0cvs20040820- 2.i586.rpm	1	0	0
libgtkhtml-2.6.1- 3.i586.rpm	1	23	0.96	libglade2-2.3.6- 3.i586.rpm	1	20	0.95
gle-3.0.6-646.i586.rpm	1	0	0	speex-1.0.4-3.i586.rpm	1	1	0.5
yast2-ncurses-2.10.5- 3.i586.rpm	1	15	0.94	perl-PDA-Pilot-0.11.8- 123.i586.rpm	1	3	0.75
cdk-4.9.10-408.i586.rpm	1	0	0	libexif-0.6.10-2.i586.rpm	1	0	0
kdebase3-kdm-3.3.0- 29.i586.rpm	1	23	0.96	libgsf-1.10.1-3.i586.rpm	1	4	0.8
kdegames3-3.3.0- 7.i586.rpm	1	22	0.96	libart_lgpl-2.3.16- 87.i586.rpm	1	0	0
liblcms-devel-1.12- 57.i586.rpm	1	0	0	libraw1394-0.10.1- 22.i586.rpm	1	0	0
yast2-qt-2.10.12- 2.i586.rpm	1	23	0.96	rekall-xbase-2.2.1- 3.i586.rpm	1	28	0.97
yast2-users-2.10.11- 2.i586.rpm	1	3	0.75	dmraid- 0.99_1.0.0rc5CDH1- 3.i586.rpm	1	1	0.5
libmal-0.31-154.i586.rpm	1	3	0.75	kdenetwork3-query-3.3.0- 35.i586.rpm	1	30	0.97
libieee1284-0.2.8- 2.i586.rpm	1	0	0				

<i>karchiver-3.0.10-28.i586.rpm</i>	1	23	0.96	<i>fontconfig-devel-2.2.96.20040728-9.i586.rpm</i>	1	2	0.67
<i>neon-0.24.7-3.i586.rpm</i>	1	3	0.75	<i>libsndfile-1.0.10-3.i586.rpm</i>	1	0	0
<i>perl-DBD-mysql-2.9004-2.i586.rpm</i>	1	2	0.67	<i>yast2-slp-2.10.0-2.i586.rpm</i>	1	6	0.86
<i>kdegraphics3-scan-3.3.0-13.i586.rpm</i>	1	36	0.97	<i>cdrecord-2.01-2.i586.rpm</i>	1	1	0.5
<i>kio_sql-0.4-216.i586.rpm</i>	1	22	0.96	<i>recode-3.6-490.i586.rpm</i>	1	0	0
<i>plptools-0.12-338.i586.rpm</i>	1	2	0.67	<i>taskjuggler-20040927_cvs-3.i586.rpm</i>	1	11	0.92
<i>gcc-3.3.4-11.i586.rpm</i>	1	0	0	<i>ldacppplib-0.0.3-28.i586.rpm</i>	1	7	0.88
<i>taskjuggler-kde-20040927_cvs-3.i586.rpm</i>	1	26	0.96	<i>openslp-1.1.5-80.i586.rpm</i>	1	3	0.75
<i>kbarcode-1.8.0-3.i586.rpm</i>	1	28	0.97	<i>libcroco-0.5.1-3.i586.rpm</i>	1	2	0.67
<i>libdar-2.1.4-2.i586.rpm</i>	1	3	0.75	<i>perl-DBD-mysql-2.9004-2.i586.rpm</i>	1	2	0.67
<i>tse3-0.2.7-214.i586.rpm</i>	1	2	0.67	<i>cracklib-2.7-1008.i586.rpm</i>	1	0	0
<i>wv2-0.2.2-3.i586.rpm</i>	1	6	0.86	<i>ksteak-0.9.4-259.i586.rpm</i>	1	30	0.97
<i>kdegraphics3-tex-3.3.0-13.i586.rpm</i>	1	25	0.96	<i>perl-DBI-1.43-2.i586.rpm</i>	1	0	0
<i>libmng-devel-1.0.8-3.i586.rpm</i>	1	2	0.67	<i>gltt-2.5.2-467.i586.rpm</i>	1	1	0.5
<i>libselinux-1.16-3.i586.rpm</i>	1	0	0	<i>kpsion-0.12-338.i586.rpm</i>	1	25	0.96
<i>xfspgrog-2.6.13-2.i586.rpm</i>	1	1	0.5	<i>kdegraphics3-pdf-3.3.0-13.i586.rpm</i>	1	24	0.96
<i>sqlite-2.8.15-3.i586.rpm</i>	1	1	0.5	<i>libstdc++-devel-3.3.4-11.i586.rpm</i>	1	0	0
<i>apache2-mod_php4-4.3.8-8.i586.rpm</i>	1	4	0.8	<i>fontconfig-devel-2.2.96.20040728-9.i586.rpm</i>	1	2	0.67
<i>frozen-bubble-1.0.0-332.i586.rpm</i>	1	11	0.92	<i>kbiff-3.6.3-332.i586.rpm</i>	1	22	0.96
<i>libksba-0.9.8-3.i586.rpm</i>	1	0	0	<i>freetype-1.3.1-1157.i586.rpm</i>	1	0	0
<i>php4-mysql-4.3.8-8.i586.rpm</i>	1	2	0.67	<i>kdeaddons3-kontakt-3.3.0-10.i586.rpm</i>	1	26	0.96
<i>pwdutils-2.6.90-6.i586.rpm</i>	1	12	0.92	<i>perl-DBI-1.43-2.i586.rpm</i>	1	0	0
<i>libdvdread-0.9.4-147.i586.rpm</i>	1	0	0	<i>libjpeg-6.2.0-736.i586.rpm</i>	1	0	0
<i>bluez-libs-2.10-2.i586.rpm</i>	1	0	0	<i>libtiff-3.6.1-47.i586.rpm</i>	1	1	0.5
<i>device-mapper-1.00.19-3.i586.rpm</i>	1	0	0	<i>libpcap-0.8.3-3.i586.rpm</i>	1	0	0
<i>freelut-devel-2.2.0-82.i586.rpm</i>	1	4	0.8	<i>libstdc++-3.3.4-11.i586.rpm</i>	1	0	0
<i>libkpi-0.0.cvs20040920-3.i586.rpm</i>	1	22	0.96	<i>libnscd-1.0-2.i586.rpm</i>	1	0	0
<i>xbsql-0.11-102.i586.rpm</i>	1	3	0.75	<i>tk-8.4.7-3.i586.rpm</i>	1	0	0
<i>freetype2-devel-2.1.9-3.i586.rpm</i>	1	0	0	<i>libzio-0.1-4.i586.rpm</i>	1	0	0
				<i>imlib2-1.1.1-2.i586.rpm</i>	1	2	0.67

yast2-xml-2.10.1-2.i586.rpm	1	3	0.75	expat-1.95.8-2.i586.rpm	1	0	0
libidn-0.5.3-2.i586.rpm	1	0	0	gawk-3.1.4-4.i586.rpm	1	0	0
sysconfig-0.31.3-17.i586.rpm	1	1	0.5	openobex-1.0.1-53.i586.rpm	1	0	0
sensors-2.8.7-2.i586.rpm	1	2	0.67	pcsc-lite-1.1.1-248.i586.rpm	1	0	0
thinkeramik-style-3.2.1-4.i586.rpm	1	11	0.92	tcl-8.4.7-3.i586.rpm	1	0	0
src_vipa-2.0.0-61.i586.rpm	1	0	0	popt-1.7-190.i586.rpm	1	0	0
libgpg-error-0.7-6.i586.rpm	1	0	0	klipsi-0.12-338.i586.rpm	1	20	0.95
libidl-0.8.3-51.i586.rpm	1	1	0.5	yast2-profile-manager-2.10.5-2.i586.rpm	1	3	0.75
libgimpprint-4.2.7-7.i586.rpm	1	0	0	gpm-1.20.1-303.i586.rpm	1	1	0.5
libsmbclient-3.0.7-5.i586.rpm	1	12	0.92	freeglut-2.2.0-82.i586.rpm	1	4	0.8
gcc-3.3.4-11.i586.rpm	1	0	0	kxmleditor-1.1.3-2.i586.rpm	1	24	0.96
linux-atm-lib-2.4.0-415.i586.rpm	1	1	0.5	perl-TermReadKey-2.21-294.i586.rpm	1	0	0
libxml2-2.6.12-3.i586.rpm	1	2	0.67	logrotate-3.7-34.i586.rpm	1	1	0.5
libxcrypt-2.2-2.i586.rpm	1	0	0	slang-1.4.9-123.i586.rpm	1	0	0
libtheora-1.0alpha3-7.i586.rpm	1	1	0.5	smpeg-0.4.5-227.i586.rpm	1	6	0.86
libstroke-0.4-737.i586.rpm	1	0	0	atk-1.6.0-4.i586.rpm	1	3	0.75
libtool-1.5.8-3.i586.rpm	1	0	0	wireless-tools-27pre26-10.i586.rpm	1	1	0.5
fvwm2-2.5.10-2.i586.rpm	1	16	0.94	boehm-gc-3.3.4-11.i586.rpm	1	0	0
hermes-1.3.2-444.i586.rpm	1	0	0	perl-Digest-SHA1-2.10-2.i586.rpm	1	0	0
pwlib-plugins-alsa-1.8.0-3.i586.rpm	1	2	0.67	libstdc++-devel-3.3.4-11.i586.rpm	1	0	0
libcap-1.92-481.i586.rpm	1	0	0	freetype2-2.1.9-3.i586.rpm	1	0	0
yast2-ldap-2.10.4-2.i586.rpm	1	9	0.9	yast2-x11-2.10.8-2.i586.rpm	1	2	0.67
yast2-packagemanager-2.10.18-2.i586.rpm	1	11	0.92	audiofile-0.2.5-41.i586.rpm	1	0	0
libacl-2.2.25-2.i586.rpm	1	1	0.5	utempter-0.5.5-2.i586.rpm	1	0	0
libattr-2.4.16-2.i586.rpm	1	0	0	parted-1.6.15-4.i586.rpm	1	2	0.67
libgcc-3.3.4-11.i586.rpm	1	0	0	perl-Digest-MD4-1.3-29.i586.rpm	1	0	0
libgcrypt-1.2.0-3.i586.rpm	1	2	0.67	yast2-sound-2.10.6-2.i586.rpm	1	3	0.75
perl-gettext-1.01-578.i586.rpm	1	0	0	aaa_base-9.2-5.i586.rpm	1	0	0
ntfsprogs-1.9.2-2.i586.rpm	1	0	0	libogg-1.1-59.i586.rpm	1	0	0
yast2-nis-client-2.10.8-2.i586.rpm	1	3	0.75	yast2-pam-2.10.3-2.i586.rpm	1	2	0.67

<i>libsamplerate-0.1.1-2.i586.rpm</i>	1	0	0	<i>itcl-3.3-419.i586.rpm</i>	2	0	0
<i>freetype2-devel-2.1.9-3.i586.rpm</i>	1	0	0	<i>libpng-1.2.6-4.i586.rpm</i>	2	0	0
<i>kdegraphics3-fax-3.3.0-13.i586.rpm</i>	1	27	0.96	<i>plptools-kde-0.12-338.i586.rpm</i>	2	24	0.92
<i>esound-0.2.35-4.i586.rpm</i>	2	2	0.5	<i>libusb-0.1.8-33.i586.rpm</i>	2	1	0.33
<i>xmms-kde-3.1-50.i586.rpm</i>	2	32	0.94	<i>OpenOffice_org-kde-1.1.3-16.i586.rpm</i>	2	25	0.93
<i>liquid-0.9.6pre4-501.i586.rpm</i>	2	18	0.9	<i>kdenetwork3-wireless-3.3.0-35.i586.rpm</i>	2	37	0.95
<i>djvulibre-3.5.14-4.i586.rpm</i>	2	12	0.86	<i>itcl-3.3-419.i586.rpm</i>	2	0	0
<i>aspell-0.50.5-40.i586.rpm</i>	2	1	0.33	<i>kdemultimedia3-video-xine-3.3.0-13.i586.rpm</i>	2	35	0.95
<i>kpl-3.1-369.i586.rpm</i>	2	24	0.92	<i>readline-5.0-1.i586.rpm</i>	2	0	0
<i>kdenetwork3-vnc-3.3.0-35.i586.rpm</i>	2	27	0.93	<i>qt3-mysql-3.3.3-24.i586.rpm</i>	2	14	0.88
<i>openct-0.5.0-3.i586.rpm</i>	2	2	0.5	<i>hwinform-9.28-2.i586.rpm</i>	2	1	0.33
<i>libxslt-1.1.9-2.i586.rpm</i>	2	1	0.33	<i>libpng-devel-1.2.6-4.i586.rpm</i>	2	0	0
<i>opensp-1.5.1-79.i586.rpm</i>	2	2	0.5	<i>gdbm-1.8.3-229.i586.rpm</i>	2	0	0
<i>kscpm-0.3-5.i586.rpm</i>	2	19	0.9	<i>gtk-1.2.10-882.i586.rpm</i>	2	3	0.6
<i>libbonoboui-2.6.1-3.i586.rpm</i>	2	38	0.95	<i>yast2-transfer-2.9.3-2.i586.rpm</i>	2	5	0.71
<i>kdebase3-samba-3.3.0-29.i586.rpm</i>	2	34	0.94	<i>kdenetwork3-IRC-3.3.0-35.i586.rpm</i>	2	22	0.92
<i>libjasper-1.701.0-2.i586.rpm</i>	2	1	0.33	<i>pcre-4.5-2.i586.rpm</i>	2	0	0
<i>pwlib-1.8.0-3.i586.rpm</i>	2	14	0.88	<i>udev-030-9.i586.rpm</i>	2	0	0
<i>kdelibs3-doc-3.3.0-34.i586.rpm</i>	2	25	0.93	<i>zlib-1.2.1-74.i586.rpm</i>	2	0	0
<i>taglib-1.3-4.i586.rpm</i>	2	1	0.33	<i>powersave-0.8.19-2.i586.rpm</i>	2	1	0.33
<i>kdegraphics3-postscript-3.3.0-13.i586.rpm</i>	2	24	0.92	<i>xorg-x11-server-6.8.1-15.i586.rpm</i>	2	7	0.78
<i>kinternet-0.66-8.i586.rpm</i>	2	26	0.93	<i>qt3-devel-3.3.3-24.i586.rpm</i>	2	11	0.85
<i>djvulibre-3.5.14-4.i586.rpm</i>	2	12	0.86	<i>OpenOffice_org-kde-1.1.3-16.i586.rpm</i>	2	25	0.93
<i>kpowersave-0.3.8-3.i586.rpm</i>	2	23	0.92	<i>openssl-0.9.7d-25.i586.rpm</i>	2	0	0
<i>xdelta-1.1.3-3.i586.rpm</i>	2	1	0.33	<i>resmgr-0.9.8-53.i586.rpm</i>	2	0	0
<i>kdemultimedia3-midi-3.3.0-13.i586.rpm</i>	2	25	0.93	<i>yast2-perl-bindings-2.10.3-2.i586.rpm</i>	2	22	0.92
<i>kdegraphics3-3D-3.3.0-13.i586.rpm</i>	2	27	0.93	<i>libgnomecanvas-2.6.0-5.i586.rpm</i>	2	22	0.92
<i>aalib-1.4.0-281.i586.rpm</i>	2	2	0.5	<i>giflib-4.1.3-4.i586.rpm</i>	2	0	0
<i>scpm-1.0-9.i586.rpm</i>	2	1	0.33	<i>cups-libs-1.1.21-5.i586.rpm</i>	2	5	0.71
<i>libavc1394-0.4.1-57.i586.rpm</i>	2	1	0.33	<i>kdegraphics3-kamera-3.3.0-13.i586.rpm</i>	2	25	0.93

<i>xbase-2.0.0-105.i586.rpm</i>	2	1	0.33	<i>synce-kde-0.8.0-2.i586.rpm</i>	3	33	0.92
<i>cdparanoia-IIIalpha9.8-546.i586.rpm</i>	2	1	0.33	<i>synce-0.9.0-2.i586.rpm</i>	3	1	0.25
<i>gpgme-0.9.0-2.i586.rpm</i>	2	1	0.33	<i>kdeadmin3-3.3.0-5.i586.rpm</i>	3	33	0.92
<i>yast2-printer-2.10.15-2.i586.rpm</i>	2	5	0.71	<i>sax2-4.8-142.i586.rpm</i>	3	11	0.79
<i>wvstreams-3.75-2.i586.rpm</i>	3	4	0.57	<i>libgnomeui-2.6.0-6.i586.rpm</i>	3	41	0.93
<i>ksimus-0.3.6-283.i586.rpm</i>	3	25	0.89	<i>xorg-x11-Mesa-devel-6.8.1-15.i586.rpm</i>	4	3	0.43
<i>libwmf-0.2.8.2-91.i586.rpm</i>	3	8	0.73	<i>flac-1.1.0-391.i586.rpm</i>	4	2	0.33
<i>libgcj-3.3.4-11.i586.rpm</i>	3	0	0	<i>libbonobo-2.6.2-3.i586.rpm</i>	4	8	0.67
<i>libvorbis-1.0.1-60.i586.rpm</i>	3	1	0.25	<i>gettext-0.14.1-33.i586.rpm</i>	4	2	0.33
<i>samba-client-3.0.7-5.i586.rpm</i>	3	14	0.82	<i>binutils-2.15.91.0.2-7.i586.rpm</i>	4	0	0
<i>openldap2-client-2.2.15-5.i586.rpm</i>	3	4	0.57	<i>kdegames3-board-3.3.0-7.i586.rpm</i>	4	25	0.86
<i>gmp-4.1.3-3.i586.rpm</i>	3	1	0.25	<i>libgnome-2.6.1-6.i586.rpm</i>	4	18	0.82
<i>gconf2-2.6.1-7.i586.rpm</i>	3	23	0.88	<i>kdeaddons3-kicker-3.3.0-10.i586.rpm</i>	4	28	0.88
<i>kdegames3-arcade-3.3.0-7.i586.rpm</i>	3	43	0.93	<i>cyrus-sasl-2.1.19-7.i586.rpm</i>	4	1	0.2
<i>rrdtool-1.0.49-2.i586.rpm</i>	3	1	0.25	<i>postgresql-libs-7.4.5-6.i586.rpm</i>	4	8	0.67
<i>plotutils-2.4.1-575.i586.rpm</i>	3	5	0.63	<i>mysql-shared-4.0.21-4.i586.rpm</i>	4	1	0.2
<i>kmldonkey-0.9.1-247.i586.rpm</i>	3	30	0.91	<i>libao-0.8.4-99.i586.rpm</i>	4	3	0.43
<i>qca-1.0-5.i586.rpm</i>	3	13	0.81	<i>OpenEXR-1.2.1-2.i586.rpm</i>	4	1	0.2
<i>hp-officeJet-0.91-123.i586.rpm</i>	3	15	0.83	<i>freeciv-1.14.2-2.i586.rpm</i>	4	36	0.9
<i>librsvg-2.6.5-3.i586.rpm</i>	3	25	0.89	<i>orbit2-2.10.5-5.i586.rpm</i>	4	6	0.6
<i>libical-0.24.RC4-41.i586.rpm</i>	3	0	0	<i>libapr0-2.0.50-7.i586.rpm</i>	4	2	0.33
<i>gail-1.6.2-3.i586.rpm</i>	3	21	0.88	<i>openjade-1.3.2-210.i586.rpm</i>	4	3	0.43
<i>glib-1.2.10-589.i586.rpm</i>	3	0	0	<i>knoda-0.7-3.i586.rpm</i>	4	28	0.88
<i>kdeutils3-laptop-3.3.0-5.i586.rpm</i>	3	23	0.88	<i>rpm-4.1.1-191.i586.rpm</i>	4	3	0.43
<i>openmotif-libs-2.2.3-6.i586.rpm</i>	3	2	0.4	<i>libapr0-2.0.50-7.i586.rpm</i>	4	2	0.33
<i>kdenetwork3-news-3.3.0-35.i586.rpm</i>	3	22	0.88	<i>amarok-1.0.2.cvs20040907-12.i586.rpm</i>	4	62	0.94
<i>capi4linux-2004.9.27-2.i586.rpm</i>	3	0	0	<i>gwenview-1.1.5-8.i586.rpm</i>	4	26	0.87
<i>thinkeramik-3.2.1-4.i586.rpm</i>	3	23	0.88	<i>alsa-1.0.6-8.i586.rpm</i>	4	0	0
<i>pilot-link-0.11.8-126.i586.rpm</i>	3	3	0.5	<i>glib2-2.4.6-5.i586.rpm</i>	4	0	0

<i>kdemultimedia3-mixer-3.3.0-13.i586.rpm</i>	5	24	0.83	<i>kdebase3-SuSE-9.2-7.i586.rpm</i>	10	36	0.78
<i>libnetpbm-1.0.0-623.i586.rpm</i>	5	0	0	<i>pango-1.4.1-3.i586.rpm</i>	11	8	0.42
<i>vdr-1.2.6-80.i586.rpm</i>	5	3	0.38	<i>imlib2-loaders-1.1.1-2.i586.rpm</i>	11	8	0.42
<i>e2fsprogs-1.35-2.i586.rpm</i>	6	0	0	<i>kdeaddons3-kate-3.3.0-10.i586.rpm</i>	11	30	0.73
<i>xorg-x11-6.8.1-15.i586.rpm</i>	6	37	0.86	<i>opensc-0.8.1-3.i586.rpm</i>	11	8	0.42
<i>ppp-2.4.2-49.i586.rpm</i>	6	3	0.33	<i>htdig-3.2.0b6-3.i586.rpm</i>	12	1	0.08
<i>yast2-storage-2.10.22-2.i586.rpm</i>	6	3	0.33	<i>koffice-presentation-1.3.3-3.i586.rpm</i>	12	55	0.82
<i>digikam-0.6.9.20040927-2.i586.rpm</i>	6	35	0.85	<i>kbear-2.1.1-44.i586.rpm</i>	12	30	0.71
<i>kdenetwork3-3.3.0-35.i586.rpm</i>	6	33	0.85	<i>kdepim3-kpilot-3.3.0-32.i586.rpm</i>	12	35	0.74
<i>jack-0.98.1-5.i586.rpm</i>	6	4	0.4	<i>htdig-3.2.0b6-3.i586.rpm</i>	12	1	0.08
<i>guile-1.6.4-127.i586.rpm</i>	6	1	0.14	<i>subversion-1.0.8-2.i586.rpm</i>	13	11	0.46
<i>swig-1.3.21-107.i586.rpm</i>	7	4	0.36	<i>k3b-0.11.15-3.i586.rpm</i>	13	47	0.78
<i>net-snmp-5.1.2-3.i586.rpm</i>	7	5	0.42	<i>pam-modules-9.2-2.i586.rpm</i>	13	6	0.32
<i>hk_classes-0.7-4.i586.rpm</i>	7	13	0.65	<i>kdepim3-organizer-3.3.0-32.i586.rpm</i>	13	44	0.77
<i>subversion-cvs2svn-1.0.8-2.i586.rpm</i>	7	23	0.77	<i>kdemultimedia3-3.3.0-13.i586.rpm</i>	14	42	0.75
<i>kdeedu3-3.3.0-6.i586.rpm</i>	7	33	0.83	<i>compat-2004.9.6-2.i586.rpm</i>	15	1	0.06
<i>kdebluetooth-0.0.cvs20040825-5.i586.rpm</i>	7	37	0.84	<i>kipi-plugins-0.0.cvs20040920-6.i586.rpm</i>	15	33	0.69
<i>heimdal-lib-0.6.2-8.i586.rpm</i>	7	6	0.46	<i>autofs-3.1.7-903.i586.rpm</i>	15	7	0.32
<i>ncurses-5.4-65.i586.rpm</i>	8	0	0	<i>gimp-2.0.4-3.i586.rpm</i>	15	42	0.74
<i>openh323-1.15.0-3.i586.rpm</i>	8	16	0.67	<i>samba-3.0.7-5.i586.rpm</i>	15	16	0.52
<i>qt3-non-qt-3.3.3-24.i586.rpm</i>	8	13	0.62	<i>arts-1.3.0-11.i586.rpm</i>	15	22	0.59
<i>db-4.2.52-90.i586.rpm</i>	8	1	0.11	<i>samba-3.0.7-5.i586.rpm</i>	15	16	0.52
<i>kdetoys3-3.3.0-5.i586.rpm</i>	9	31	0.78	<i>kdeartwork3-3.3.0-7.i586.rpm</i>	15	46	0.75
<i>kdemultimedia3-CD-3.3.0-13.i586.rpm</i>	9	35	0.8	<i>kdeaddons3-sound-3.3.0-10.i586.rpm</i>	15	65	0.81
<i>libquicktime-0.9.3-2.i586.rpm</i>	9	23	0.72	<i>xorg-x11-Mesa-6.8.1-15.i586.rpm</i>	17	4	0.19
<i>bind-libs-9.2.4-3.i586.rpm</i>	9	2	0.18	<i>kdepim3-sync-3.3.0-32.i586.rpm</i>	18	30	0.63
<i>kdebase3-SuSE-9.2-7.i586.rpm</i>	10	36	0.78	<i>yast2-core-2.10.16-2.i586.rpm</i>	18	15	0.45
<i>imlib-1.9.14-188.i586.rpm</i>	10	6	0.38	<i>mozilla-mail-1.7.2-17.i586.rpm</i>	19	1	0.05
<i>qt3-3.3.3-24.i586.rpm</i>	10	13	0.57	<i>xmms-lib-1.2.10-56.i586.rpm</i>	20	15	0.43

<i>kdemultimedia3-sound-3.3.0-13.i586.rpm</i>	20	43	0.68	<i>perl-5.8.5-3.i586.rpm</i>	49	3	0.06
<i>gnome-vfs2-2.6.1-38.i586.rpm</i>	20	31	0.61	<i>ghostscript-library-7.07.1rc1-207.i586.rpm</i>	50	12	0.19
<i>koffice-spreadsheet-1.3.3-3.i586.rpm</i>	20	40	0.67	<i>apache2-2.0.50-7.i586.rpm</i>	51	12	0.19
<i>rekall-2.2.1-3.i586.rpm</i>	21	26	0.55	<i>python-2.3.4-3.i586.rpm</i>	51	8	0.14
<i>ethereal-0.10.6-3.i586.rpm</i>	22	23	0.51	<i>apache2-2.0.50-7.i586.rpm</i>	51	12	0.19
<i>glibc-devel-2.3.3-118.i586.rpm</i>	22	3	0.12	<i>apache2-prefork-2.0.50-7.i586.rpm</i>	52	12	0.19
<i>evms-2.3.3-2.i586.rpm</i>	22	1	0.04	<i>apache2-prefork-2.0.50-7.i586.rpm</i>	52	12	0.19
<i>unixODBC-2.2.9-4.i586.rpm</i>	22	1	0.04	<i>libgphoto2-2.1.4head-5.i586.rpm</i>	54	4	0.07
<i>xorg-x11-devel-6.8.1-15.i586.rpm</i>	23	8	0.26	<i>kvirc-3.0.1-4.i586.rpm</i>	54	41	0.43
<i>xorg-x11-devel-6.8.1-15.i586.rpm</i>	23	8	0.26	<i>xmms-plugins-1.2.9-68.i586.rpm</i>	59	42	0.42
<i>perl-Tk-800.024-430.i586.rpm</i>	23	1	0.04	<i>sane-1.0.14-5.i586.rpm</i>	61	34	0.36
<i>kdeutils3-3.3.0-5.i586.rpm</i>	24	29	0.55	<i>mozilla-1.7.2-17.i586.rpm</i>	79	41	0.34
<i>kdetv-0.8.2-4.i586.rpm</i>	24	34	0.59	<i>iptables-1.2.11-4.i586.rpm</i>	80	1	0.01
<i>kdegraphics3-3.3.0-13.i586.rpm</i>	24	39	0.62	<i>ImageMagick-6.0.7-4.i586.rpm</i>	99	12	0.11
<i>kdemultimedia3-video-3.3.0-13.i586.rpm</i>	26	50	0.66	<i>kdepim3-3.3.0-32.i586.rpm</i>	99	43	0.3
<i>koffice-1.3.3-3.i586.rpm</i>	28	32	0.53	<i>kdelibs3-3.3.0-34.i586.rpm</i>	115	54	0.32
<i>gtk2-2.4.9-10.i586.rpm</i>	28	19	0.4	<i>glibc-locale-2.3.3-118.i586.rpm</i>	191	0	0
<i>zsh-4.2.1-2.i586.rpm</i>	29	1	0.03	<i>xine-lib-0.99.rc6a-4.i586.rpm</i>	209	16	0.07
<i>koffice-illustration-1.3.3-3.i586.rpm</i>	30	39	0.57	<i>kdebase3-3.3.0-29.i586.rpm</i>	213	70	0.25
<i>DirectFB-0.9.20-88.i586.rpm</i>	30	9	0.23	<i>wine-20040813-7.i586.rpm</i>	233	9	0.04
<i>kdeaddons3-konqueror-3.3.0-10.i586.rpm</i>	31	56	0.64	<i>wine-20040813-7.i586.rpm</i>	233	9	0.04
<i>glibc-2.3.3-118.i586.rpm</i>	31	0	0	<i>OpenOffice_org-1.1.3-16.i586.rpm</i>	250	28	0.1
<i>pam-0.77-227.i586.rpm</i>	36	2	0.05	<i>OpenOffice_org-1.1.3-16.i586.rpm</i>	250	28	0.1
<i>koffice-wordprocessing-1.3.3-3.i586.rpm</i>	40	49	0.55				
<i>kdenetwork3-InstantMessenger-3.3.0-35.i586.rpm</i>	41	46	0.53			Average	0.48
<i>xorg-x11-libs-6.8.1-15.i586.rpm</i>	47	4	0.08				