



Reasoning® Inspection Service

Reasoning
Inspection Service
Defect Metrics

Apache
2.1-dev
OPEN SOURCE

30-Nov-2004

Discovery Mapping Analytics™ is a mark of:



P.O. Box 478

Menlo Park, CA 94026-0478

+1 650-324-2510

www.reasoning.com

INTRODUCTION

Reasoning™ Inspection Service

Reasoning Inspection Services for Java, C, and C++ provide essential expertise that boosts the productivity of development teams by finding software defects faster, earlier, and at a far lower cost than traditional approaches. The Reasoning service provides many of the benefits of a manual code review, but in significantly less time and at a dramatically lower cost. Reasoning detects and diagnoses defects well before they become discernible problems and provides a roadmap to the exact location for remedy and resolution.

Reasoning achieves these ends by automating software inspection, which enhances the organization's existing QA processes and bolsters efforts to provide easy-to-support, high-quality software. As a result, Reasoning enables fast time to ROI and permits software developers to focus on their core competency - software development.

Deliverables

Reasoning provides two types of reports at the conclusion of the inspection process:

- The Defect Data report which makes defect analysis and repair simple by identifying the type and location of every defect and describing the circumstances under which they will occur. By providing this map to discovered defects, development time can be spent more effectively.
- The Defect Metrics report is designed for development managers. Providing a better insight into to problem areas within an application, it allows managers to better plan testing and development efforts.

This is the Defect Metrics report.

Application Overview

Reasoning inspected 360 user files in the Apache 2.1-dev 1/31/2003 code, including all the source files.

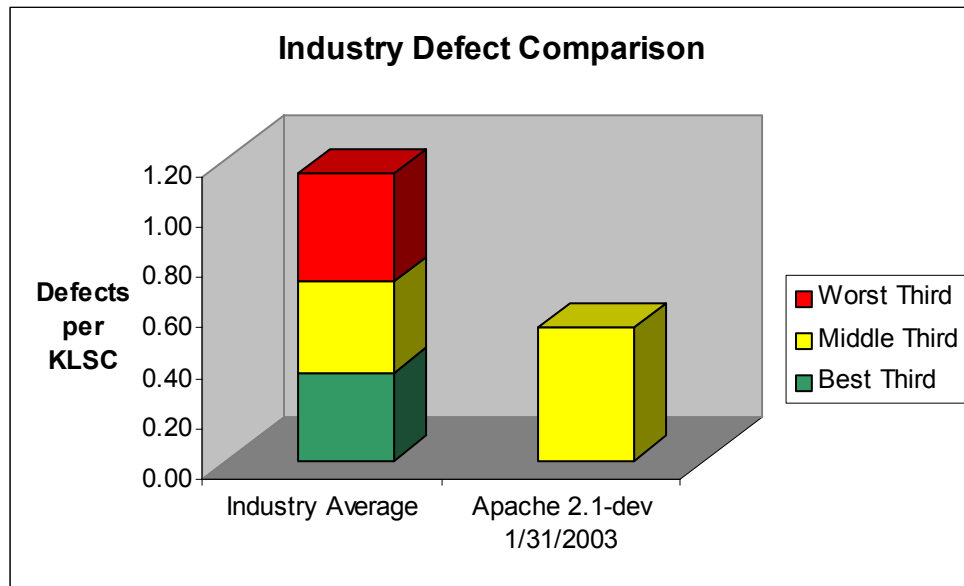
Application overview

Total Number of Source Files:	169
Number of User Include Files:	191
Total Number of User Files Processed:	360
Total Lines of Source Code in Source Files (LSC):	58,944
Number of Lines in User Include Files:	17,264
Total Lines of Code in Project:	76,208

REASONING METRICS

Industry Comparison

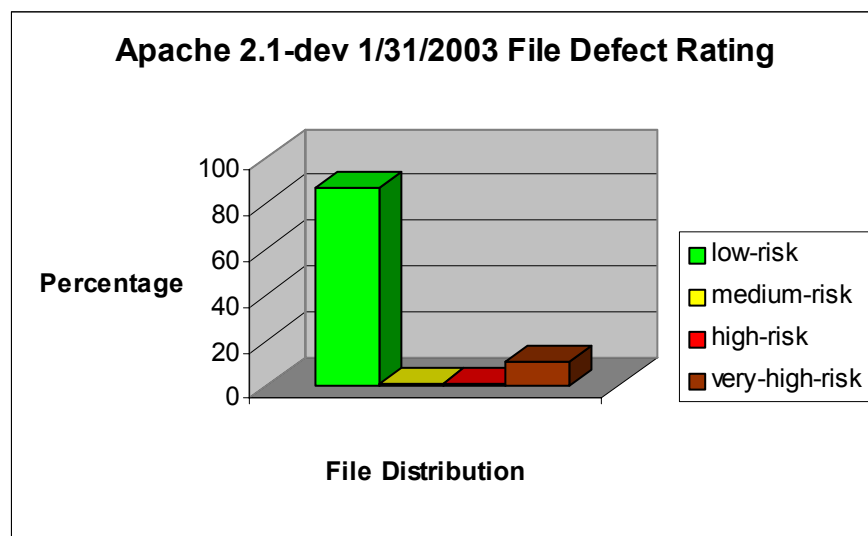
Reasoning found 31 defects in 58,944 source lines, a defect density of 0.53 Defects/KSLC. In a sampling of 200 projects totaling 35 million lines of code, 33% had a defect density below 0.36 Defects/KSLC (green), 33% had a defect density between 0.36 and 0.71 Defects/KSLC (yellow), and the remaining 33% had a defect density above 0.71 Defects/KSLC (red).



File Defect Rating

The following graph shows the percentages of files with low, medium, high and very high defect densities, defined with respect to the average defect density of this application:

- *Low* is less than 0.5 times the average defect density;
- *Medium* is between 0.5 and 1.0 times the average defect density;
- *High* is between 1.0 and 2.0 times the average defect density;
- *Very high* is higher than 2.0 times the average defect density.



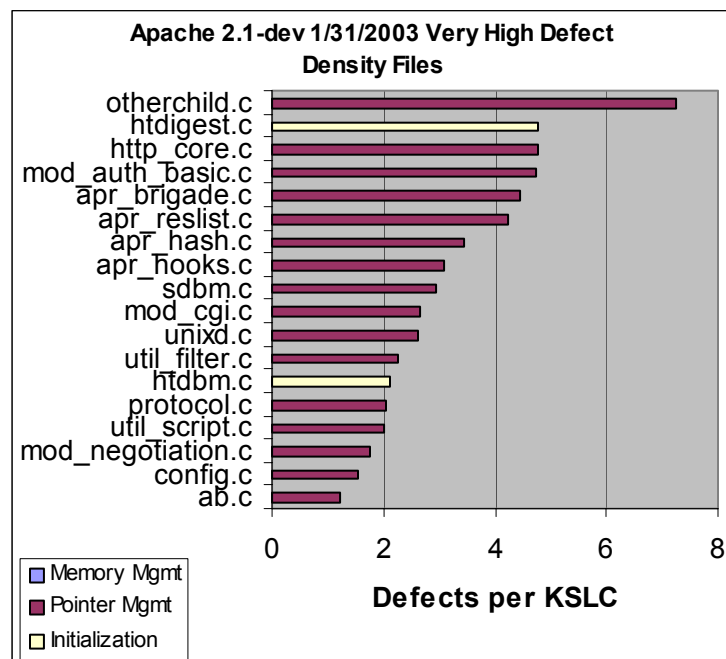
This graph shows how defects are distributed across the files. Some files may exhibit unusually high defect densities, rendering them more likely to fail in the field. Such files may require more testing or rewriting.

Very High Defect Density Files

In the Apache 2.1-dev 1/31/2003 code, 18 files had a defect density in the very high range. The graph below shows the defect density in descending order for these files. It also shows how those defects are distributed over defect classes.

The defects contained in each defect class are listed below:

Defect Class	Defect
Initialization	Uninitialized Variable
Pointer management	NULL Pointer Dereference
	Out of Bounds Array Access
Memory management	Memory Leak
	Bad Deallocation



Defect Summary

The column *Defect Instances* in the table below details, per defect class, how many defects there are in the application.

The column *Files Affected* details, per defect class, the number of files in the application that have one or more defects. This is an indication of how much work is needed to fix the defects; many defects in one file are easier to fix and retest than the same number of defects scattered over many files.

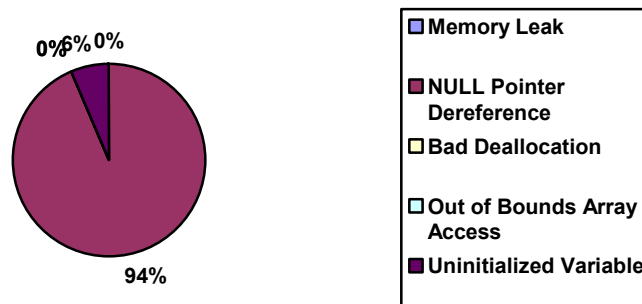
Defect Summary

Inspection Class	Defect Instances	Files Affected
Memory Leak <i>Reference to allocated memory is lost</i>	0	0
NULL Pointer Dereference <i>Expression dereferences a NULL pointer</i>	29	20
Bad Deallocation <i>Deallocation is inappropriate for type of data</i>	0	0
Out of Bounds Array Access <i>Expression accesses a value beyond the array</i>	0	0
Uninitialized Variable <i>Variable is not initialized prior to use</i>	2	2
Total Defect Instances	31	--

Defect Distribution

The graph below shows how the defects in the entire application are distributed over the five types of defects

Defect Distribution





P.O. Box 478

Menlo Park, CA 94026-0478

+1 650-324-2510

www.reasoning.com