2011

The Relational Database: a New Static Analysis Tool?

Adam M. Dutko
Cleveland State University

THE RELATIONAL DATABASE: A NEW STATIC ANALYSIS TOOL?

ADAM M DUTKO

Bachelor of Chemistry

Davidson College

August, 2003

submitted in partial fulfillment of the requirements for the degree

MASTERS OF SCIENCE IN SOFTWARE ENGINEERING

at

CLEVELAND STATE UNIVERSITY

July, 2011

This thesis has been approved for the

Department of **ELECTRICAL AND COMPUTER ENGINEERING**

and the College of Graduate Studies by

_____

Thesis Committee Chairperson, Dr. Nigamanth Sridhar

_____

Department/Date

_____

Dr. Yongjian Fu

_____

Department/Date

_____

Dr. Wenbing Zhao

_____

Department/Date

Redbird and Bean

THE RELATIONAL DATABASE: A NEW STATIC ANALYSIS TOOL?

ADAM M DUTKO

ABSTRACT

Code comprehension is pivotal to reducing errors in software. Reading source code improves code comprehension and enables effective fixes but as a code base grows meta-data become increasingly important. Static Analysis techniques provide an avenue for software developers to learn more about their code through meta-data while also helping them safely detect potential errors in their source. Unfortunately, many Static Analysis tools have a steep learning curve and are limited in scope. This thesis seeks to make Static Analysis accessible and extensible by asking what ubiquitous tools like SQL and relational databases can offer and what they cannot. We begin to answer these questions by exploring the source code of three C++ projects (libodbc++, log4cxx, C++ Sockets Library) using a new Static Analysis tool called Trike. Initial results indicate Trike is a promising and accessible tool for analyzing the structure of a code base. With further improvements, Trike should equal more established Static Analysis tools in scope and surpass them in usability.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

Introduction

Software was once considered a luxury because of its high cost and the relative scarcity of systems to run it on [4, 29, 46], Some of the earliest adopters were banks and researchers [4, 29]. As these groups illustrated the utility of leveraging software to perform calculations and the cost of computers decreased, other groups began to push for the development of new software packages [4, 29]. This demand helped create a vibrant industry that today employs and services a large percentage of the world's population [4, 46].

When society relied less on software based systems, downtime was an inconvenience. Now that society depends on software based systems, errors are much more than a simple inconvenience [4, 20]. Our dependency on software makes writing correct and error free software essential to commerce and critical to humanity. So how is it that we find it so hard to eliminate errors in software [4, 20]? A consensus on why this is so difficult remains elusive. Ensuring software is free from errors has become the quintessential goal of the field of *Software Engineering* [46]. Learning how to reduce errors in software is a facet of this pursuit.

My thesis seeks to make existing error elimination techniques easier to use by leveraging a technology familiar to most developers: the relational database. I explore this new approach to the field of *Static Analysis* using the first SQL based

Static Analysis tool called Trike. The methods introduced by Trike, through the exploration of real C++ source code, should make it easier for software developers to extract meaningful meta-data from their C++ projects to help them understand how projects change over time. I believe this introduction serves as a good starting point for research into SQL based Static Analysis and defend this belief by discussing why the extracted meta-data should motivate developers to explore this new direction in the field of *Static Analysis*.

1.1 The Problem

Most modern software development processes provide developers with the opportunity to reflect upon errors, develop and implement patches and implement process improvements and tools to help prevent the same errors in future projects and steps in the development life-cycle [20, 46]. This approach to professional software development seems likely to persist until all software can be proven correct, in a trivial manner. Relying on software developers to discipline themselves to incorporate processes and tools into their work flow remains the only practical route to reducing errors in software [20, 29, 46]. Yet one of the highest barriers to adopting tools and processes is the level of effort required for developers to learn and integrate them into their work flow [3, 4]. If a tool or process is difficult to use it will most likely be avoided. If a tool is relatively easy to use it should have a naturally lower barrier to adoption and experience an increased level of adoption.

Each Static Analysis tool is unique and the constraints affecting the approaches tool developers take to build useful tools often elicits unique solutions. Some approaches are as unique as writing tool specific languages used to extract data and help identify errors while others are as simple as limiting the tool in scope and extensibility [5,7,8,11,13–15,19,26–28,30,32,34–40,42,43,45,57–61]. When tool developers choose to rely upon tool specific languages they make their tool less appealing to busy developers and when they reduce the scope of their tool too much, the effort of using their tool eclipses any derived benefits. Without approachable and useful tools many

software developers will avoid the pain associated with integrating a Static Analysis tool into their work flow. This is something that Trike seeks to change for C++ developers.

1.2 The Thesis

The central question I seek to answer in my thesis is if a relational database can be used for Static Analysis. I arrived at this question because I wanted to make *Static Analysis* approachable to software developers and easy to integrate into their work flow. In my mind, the most effective way to accomplish these goals is to identify a seemingly unrelated tool with similar traits, such as a relational database, and then use it to build an approachable and powerful Static Analysis tool.

1.3 The Solution Approach

Before I started development on Trike, I explored existing Static Analysis tools and discovered that many of them were either expensive, complex, incomplete or a mixture of the three. Trike does not require a costly license, something that is easily accomplished through code reuse and extension using code from liberally licensed projects like GCC-XML and by making the interface for extending Trike as easy as writing a new SQL query. Trike is not complex, something that is accomplished by making the analysis language the same language in use by most developers to consume data from a database, SQL. Trike is capable, a tool that is currently able to extract useful meta-data about C++ projects.

I believe my solution approach addresses the three important issues affecting the adoption and use of Static Analysis tools today: cost, complexity and flexibility. Trike does not eliminate these issues but serves as a proof of concept showing the potential for using the relational database model to perform Static Analysis on C++ code. Some work remains to grow Trike to encompass other areas of Static Analysis. In the future, I might have to replace GCC-XML with another tool like CIL or make

modifications to GCC-XML to accommodate such enhancements but for now GCC-XML provides a useful starting point.

1.4 Contributions

My thesis asks if a relational database can be used to perform Static Analysis. I begin to answer this question by building a prototype of the first Static Analysis tool to leverage the relational database, called Trike. I then use Trike to show the expressiveness and utility of leveraging a relational database to gather structural meta-data on three different C++ programs (libodbc++, log4cxx, the C++ Sockets Library). In order to implement Trike, I had to write a GCC-XML wrapper capable of generating whole C++ program XML representations from individual source files and a unique database schema based on the Extensible Markup Language (XML) files generated by GCC-XML and constrained by an experimental Document Type Definition (DTD) for C++.

Overall, the implementation of Trike resulted in

- a wrapper program used to leverage existing code and technologies,
- a unified configuration file with settings for three example projects,
- an experimental C++ language database schema,
- a utility for generating and importing XML representations of C++ source code,
- queries used to illustrate the usefulness of Trike for three different C++ programs,
- and the implementation of the first *Static Analysis* tool to leverage a relational database.

1.5 Organization of the Thesis

My thesis has five major sections. The first section provides an "Introduction" and has five subsections that I use to state why reducing errors in software is important, ask if it is possible to use a relational database to statically analyze software, discuss my approach to answering that question, and outline my contributions to the field of *Static Analysis*. I use the second section to discuss "Related Work" in the field of Static Analysis as well as discuss relevant work related to relational databases and extensible markup language (XML). I use the third section to discuss the "Trike Architecture," which includes discussion about the relevant sub-components, GCC-XML, the XML schema, the database schema and my approach to mapping XML to tables, using the SQL data definition language. I then use the next section, Section four, to discuss the bulk of my work. In this section I discuss Static Analysis from the context of Trike, identify three relevant C++ projects and discuss the results of the structural, composite and advanced queries I ran on their respective code bases. I conclude the fourth section with a discussion about why developers should find the results from these queries intriguing. Finally, I use the fifth section to discuss the potential limitations of Trike and areas where Trike could improve.

CHAPTER II

Related Work

It is surprising not to discover other researchers utilizing a database to perform static analysis on a code base. Other researchers have explored XML then used XQuery to check constraints while others have used XML as a means of generating an AST and analyzed the code base with programmable constraints [8, 47, 61]. To my knowledge, the use of a database and SQL to perform static analysis is unexplored until now. Trike is a new Static Analysis framework currently capable of extracting useful project meta-data. Trike is not at the level of other existing tools like the Clang Static Analyzer, cppcheck, cpplint and the Oink stack but with more work it should be and should have the added benefit of being accessible to beginning developers as well as being faster at regenerating the model on which query results are based [10,25,33,56].

2.1 Static Analysis

Static Analysis is a conflated term and serves as an umbrella for the varied approaches to analyzing source code before it is executed [16]. Approaches to static analysis vary because not all languages are the same [16]. Some languages are strongly typed while others are weakly typed. Some languages support automatic memory management while others require manual memory management. Some language compilers gen-

erate byte code and operate atop a virtual machine while others generate machine code and operate directly atop the hardware. These disparities influence the possible errors that can exist in a particular base of code which when coupled with the syntactical differences between languages makes writing a single static analysis tool for all languages and errors, difficult, if not impossible.

Static Analysis techniques help developers gain a better understanding of their program by revealing important structural meta-data and detecting potential errors before they become larger problems. Using Static Analysis techniques, developers can discover errors that do not depend on values of variables and objects such as semantic errors, type errors, memory errors, logic errors, interface and include errors and various security errors associated with each [16]. Developers can also learn important structural data about their programs such as line count, character count, file count, class composition, and function count. Such meta-data not only enhance code comprehension but can also prove useful during code refactoring when generating code statistics becomes important. Some useful statistics that can enhance the process of code refactoring include the total lines of code for a project, the average lines of code per file, the number of functions per file, the number of classes per file, functions per line of code (or function density), classes per line of code (or class density), and the total number of system and library files used [16].

Static Analysis tools typically perform these operations using program models [16]. Such models vary with what data the tool has available, what errors and meta-data a tool is built to discover and the source language of the program(s) to be analyzed [16]. In order to construct the model, a Static Analysis tool must typically have access to the source code, the Abstract Syntax Tree (AST), the byte code or the compiled binary [16]. Although most tools rely on a model to analyze a program some tools do not require a model.

If the tool is built to discover errors, it might use language constructs (eg. variables, functions) or symbols to build a model such as a well connected graph of execution (with or without contextually sensitive data), then use the relevant aspects of the graph and tool specific logic to identify potential errors [16]. If the

tool is built to discover meta-data like structural information it might use the same language constructs but bypass model generation and simply calculate each relevant data point using heuristics or logic. If the tool is built to extract meta-data and identify potential errors it might take a divide and conquer approach, generating a model when it needs it and using heuristics and logic when it does not.

Trike is written to statically analyze C++ programs and requires the source code of the target program. The source code is required because Trike uses it as the basis for generating an XML model for each file which provide the basis for static program analysis in Trike. After a model is generated, it is parsed and loaded into the database element by element. The developer then writes SQL queries to perform static program analysis on their code.

A myriad of Static Analysis tools exist in the literature and industry. It is remarkable that none of the existing static analysis tools utilize a database and SQL to analyze source code. Of the tools that exist I chose to compare the Clang Static Analyzer, cppcheck, cpplint and the Oink stack to Trike.

The Clang Static Analyzer is different from Trike for a few reasons. One of the most important is that it is targeted to discovering bugs in C and Objective C programs. The project is implemented in C++ but C++ is not currently a language targeted for analysis despite there being support for C++ in its tool-chain. Support for it should be added eventually and might already be ready to commit but no such support is listed on the homepage. In addition to this difference, the Clang Static Analyzer can either be run as a stand-alone tool or from within the Apple Computer Integrated Development Environment called Xcode [1, 33]. Trike is currently only a stand-alone tool. Another important difference is that the Clang Static Analyzer is restricted to performing a particular set of static analysis routines implemented in the program logic. In order to extend Clang a patch must be submitted [33]. The Clang Static Analyzer is not completely different from Trike. The Clang Static Analyzer also uses a compiler infrastructure and requires source code. Instead of using the GNU Compiler Collection (GCC), like Trike, it uses the Low Level Virtual Machine (LLVM) infrastructure to analyze source code. It is also open source, just like Trike.

cppcheck is another useful static analysis tool. cppcheck focuses on discovering a variety of well-known errors as well as adherence to some industry accepted coding guidelines in C and C++ code. Some of the errors and guidelines cppcheck looks for include checking for the use of auto variables, bounds checking, class attributes, exception safety, memory leaks, null pointers, obsolete functions, standard template library usages, uninitialized variables, unused functions and using postfix operators. While these are all useful errors and guidelines to check for the approach used by cppcheck is different from Trike. cpplint uses an executable with a limited set of predefined rules managed by the cppcheck developers. Like the Clang Static Analyzer, cppcheck can be integrated into an IDE such as Microsoft Visual Studio or the Eclipse Foundation called Eclipse [10,48]. There are also cppcheck plugins for several Continuous Integration tools like Hudson and Jenkins. cppcheck can be extended by modifying the code which just like Trike, is also open source.

cpplint, the tool released by Google, is a very light static analysis that mainly focuses on identifying code that does not adhere to the Google C++ style guide [25]. cpplint focuses on syntax errors and best practices according to Google. It performs these duties using various heuristics on source code and is open source like Trike. It is also written in Python, just like Trike.

Of all the tool surveyed, the most advanced is the Oink stack [56]. The Oink stack is "a collaboration of C++ static analysis tools" and utilizes the CQual solver as the basis for its CQual++ polymorphic whole-program data-flow analysis engine. Like Trike, Oink uses GCC as part of its tool stack. Oink is able to check for syntax errors and for the use of best practices defined by the developers. Oink is also extensible and composable enabling developers to add their own rules for analysis and add it to their own projects. Most importantly, Oink is able "to compute expression-level and type-level data-flow and statement-level intra-procedural control-flow" which enables advanced static analysis techniques like data-flow analysis to be explored. Oink is also open source, just like Trike.

Other Static Analysis tools that exist in industry were not explored because of their high licensing cost and because many of the useful comparisons like runtime,

the number of false positives and coverage are not yet relevant because Trike is a prototype and currently limited to structural static analysis.

2.2 Relational Databases

A relational database enables developers to express relationships between sets of data. Mathematical set theory provides the basis for building these relationships while the SQL language provides the means for conveying them in a structured and consistent manner [9]. These constructs provide the foundation for the logic associated with the storage, modification and retrieval of arbitrarily large amounts of data [9]. Such data are stored in tables, with each row of a table containing a unique record comprised of the data needed to describe the entity data stored in the row, partitioned into columns. Each row is identified with a unique id "(known as the *primary key*)" that helps distinguish it from other entities in the table. The *primary key* forms the foundation for building relationships across tables relating entities to other entities [2].

The tables used to store the data in a database are created using SQL schema statements, data is retrieved and modified in the database using SQL data statements and transactions are initiated, committed and rolled back using SQL transaction statements. Many implementations of these technologies exist but the one I used is a branch of the relational database MySQL called MariaDB. "MariaDB is a database server that offers drop-in replacement functionality for MySQL." [55] It is a relative newcomer to the field of relational databases but is based on the very well established code base of MySQL. MariaDB strives to maintain compatibility with MySQL, now owned by Oracle Corporation [41], but also seeks to implement new and interesting features useful to developers.

2.3 Extensible Markup Language

The initial purpose behind the development of Extensible Markup Language (XML) was to create a portable platform agnostic representation of structured data [6,12,51].

XML is now an established technology used by organizations around the world [51]. Beyond this original hierarchical design of variably nested empty and non-empty elements, XML enables developers to construct their own rule-sets referred to as schemas, that enforce the structure of data stored in XML files. Established tools like XPath help developers traverse documents using root nodes and more recent tools like XQuery enable developers to combine XML documents with other documents to derive new output [52, 53]. XML even has its own transformation language called XSL Transformations (XSLT) that empowers developers to change XML documents into other types of XML documents such as HTML, XHTML or SVG as well as add presentation enhancements [54]. The real power behind XML with regard to Static Analysis is that it provides an implicit model for explaining the relationship between source code elements. GCC-XML generates XML documents using the intermediate representation of C++ source code and as a result provides Trike with the implicit source element relationships of a project.

CHAPTER III

Trike Architecture

Trike is the name given to the code performing the work outlined in this Thesis. The name "Trike" is a colloquialism for the word tricycle and was chosen because many people first ride a tricycle as a child. It is hoped that Trike will serve a similar purpose with regard to *Static Analysis*. Using a new approach to established techniques, developers should be able to use Trike to improve their C++ code base. Trike is written in the Python language and provides a scaffolding around the individual components used in this Thesis, such as GCC-XML and MariaDB, to create a cohesive and usable tool from disparate components.

Trike generates XML using GCC-XML so the source code it analyzes needs to be capable of being built by the GCC tool-chain. Trike provides the logic and data to GCC-XML required to generate XML representation of source files. The provided logic detects source files using the extensions specified in the Trike configuration file with a simple pattern matching algorithm. Once the list of valid files is generated, Trike then calls GCC-XML for each file from the list and generates an XML representation of the source based upon the intermediate representation created by GCC. The scaffolding Trike provides to MariaDB consists of the logic used to process the schema definition file specified in the Trike configuration file, logic to apply the statements to the database and logic to ultimately generate and run queries requested by

other portions of the Trike code base. Trike also provides the logic for generating queries to upload the entire contents of the original source files and their generated XML files, provides the logic for parsing the generated XML using the Simple API for XML (SAX) and includes the logic for loading the generated data into the database. In the future, Trike will also provides a utility to regenerate XML representations of modified source files and replace their data in the database.

3.1 The GCC-XML Extension

The GNU Compiler Collection (GCC) implements multiple front-ends in its mainline source tree used to transform raw source code into abstract trees which can be transformed, through lower level processes, into machine code and ultimately into an executable [49]. The C++ front-end to GCC implements ISO C++ and is able to parse whole source trees [49]. The purpose of the C++ front-end is to generate consumables for other parts of the compilation process [49]. These efforts are naturally not geared towards generating alternative representations of source code. For instance, if a developer wanted to perform whole C++ program parsing and generate a program representation in extensible Markup Language (XML) they would have to create a wrapper, or quite possibly an extension to take the intermediate representation of the parsed C++ source and generate XML.

This is not a trivial process, which is quite possibly why few tools exist to perform this function [24, 40]. GCC-XML is one of the few tools that can generate XML from C++ source code. GCC-XML leverages the C++ front-end to GCC which is important as GCC is able to deal with the entire C++ language specification [24]. It is important to make note that GCC-XML is oriented to generating XML representations for each individual source file. This means that when it generates XML for each individual source file, it is prone to generating redundant information. Although, this is ideal for development tools like Integrated Development Environments (IDE) which need to convey source code in a tangible and useful way to developers on a per file basis, it is not ideal for tools like Trike. Development environments often make

use of visual elements to enable developers to collapse and expand functions to save space on a page and enhance understanding of a code base, to highlight variables, types and other syntactically relevant elements in different colors to deepen understanding of the code displayed on the screen, while Trike is interested in generating whole program representations using non-redundant elements. This augmentation to Trike is still under development.

Despite this fleeting shortcoming, the utility of GCC-XML did not escape me, as it makes whole program representation easy and partial program analysis accessible. Before GCC-XML generates XML it creates an intermediate representation with all includes resolved and imported. Once this step is complete it generates the XML representation of the code and saves it as specified by the user. Trike calls GCC-XML for each valid source file and saves the generated XML using the original filename but with an XML extension instead of the original extension. An alternative approach would have been to generate the intermediate representation of the source code for the entire program, save it, then call GCC-XML on this file, which in the long run might lead to less elemental redundancy in the database. This is especially true with respect to maintaining proper namespace for portions of code. It is acknowledged that the whole intermediate approach might have been superior to the one chosen but the current approach will enable Trike to selectively regenerate XML representations which when coupled with logic to detect existing elements should enable fast and efficient partial program analysis after developers make minor changes.

While building Trike, several assumptions were made. The most important assumption, which was verified before running the tool, was that the source code given to Trike builds with few warnings and no errors. Another was that the project contained easily discoverable include directories where header files and other important portions of code were accessible. The last, and most important, assumption was that GCC-XML was bug free to the point of being usable on a very large scale.

The last assumption merits further discussion as it became apparent that when analyzing large projects GCC-XML works to the point the code compiles. This operation can take quite a long time which on slower machines can present the user

with no way to monitor the progress or track errors. I also suspect GCC-XML could be improved by adding logic to capture important components of the code base like function bodies which would enhance the analysis capabilities of Trike. To accommodate generated warnings, Trike includes logging to a database table as well as logging of all operations to a log file. For longer builds, Trike also includes a progress indicator to help indicate the progression throughout the XML generation process. If the generation process is interrupted, by the user or by an error, the project must be deleted, and all data reprocessed. Future releases should make this scenario less painful by relying on the innate capability of GCC to selectively generate representations of files that are missing or modified.

The progress indicator and logging will prove useful in future work when we being to compare Trike to competing *Static Analysis* tools. The file counter currently indicates how many files have generated out of the total and the log currently includes time-stamps so the user can begin to estimate how long each run should take. These elements are essential for developers working under time constraints as they will help them appropriately schedule their builds. Trike also indicates which files failed to generate and provides the user with a list in addition to the STDERR contents in a database table. An added benefit of cataloging these messages is that some of them might correspond to valid compiler failures. Even if they represent false positives they are cataloged and are not presented in a manner that would confuse developers or distract them from their main purpose. With tools like Trike, having false positives is often better than false negatives [16]. I believe the current logging approach balances utility with the responsibility of reporting potential errors that in most environments should be appreciated. I also believe logging in Trike should be more granular but that the current level of logging is commensurate with the current capabilities of Trike.

3.2 The XML Schema

The Document Type Definition (DTD) contributed by the GCC-XML developers was used to determine what language elements were currently mapped from C++ to XML by GCC-XML [21–23]. The rules contained in the definition were used to ascertain which elements required which attributes and to a limited extent what type of data each attribute should contain. I am aware the language elements mapped in the DTD might not contain every possible aspect of the C++ grammar because GCC-XML is based on GCC. I am also aware the current DTD is an amalgam of efforts by volunteer contributors to GCC-XML and that the authors of GCC-XML acknowledge it is still a work in progress [21–23].

Today, GCC-XML supports many useful components of the C++ language. The language elements and related meta-data elements that are currently mapped include argument, arraytype, base, class, constructor, converter, cvqualifiedtype, destructor, enumeration, enumvalue, field, file, function, functiontype, fundamentaltype, method, methodtype, namespace, namespacealias, offsettype, operatorfunction, operatormethod, pointertype, referencetype, struct, typedef, unimplemented, union, and variable. I believe unimplemented serves as a "catch-all" for elements not currently supported by GCC-XML. Because GCC-XML is still under development, the XML schema was not used to actively validate the generated XML. However, with some changes and further inspection of GCC-XML, using the schema for such verification could prove useful to ensure the integrity of generated data to a particular C++ dialect and might lead to a beneficial comparison to existing schema query tools [61].

3.3 The Database Schema

The database holds all of the data generated by Trike with the exception of the data generated by the log file and the settings stored in the configuration file. The underlying database schema is essential to the organization of this data and a logical requirement for building relationships about various aspects of the code base to realize

the utility of Trike as a *Static Analysis* tool. In order for developers to harness the data generated by Trike, each XML element needs to be properly mapped to a table and the element attributes to columns in the table. Work on mapping the hierarchy of elements (and their attributes) to a database exists in the literature and was used as the basis for my approach [6,12,17,18,31,47,50]. In total, 30 tables exist that map the 30 XML elements present in the XML files generated by GCC-XML. Each table in the schema is named with a prefix of **gccxml_** then the name of the element. The one exception to this is the table for the root element which is simply named **gccxml**. The additional five tables that exist in the database are for storing useful meta-data about the project such as **errors** table for holding GCC-XML errors related to the project, the **program** table for holding meta-data about the project, the **file** table for holding meta-data about source and XML files, the **source** table for holding a reference to a valid source file in the **file** table and the actual source from the file, and finally the 'xml' table for holding a reference to a valid xml file in the 'file' table and the actual xml from the generated xml file. The following graphic shows the relationships between some of the tables in the database for the C++ Sockets Library project and was generated using SchemaSpy [44]. An alternative, and complete listing of tables is shown in 2 the "Show Tables" figure after the partial representation shown in 1.

Figure 1: Example Trike Database Relationships

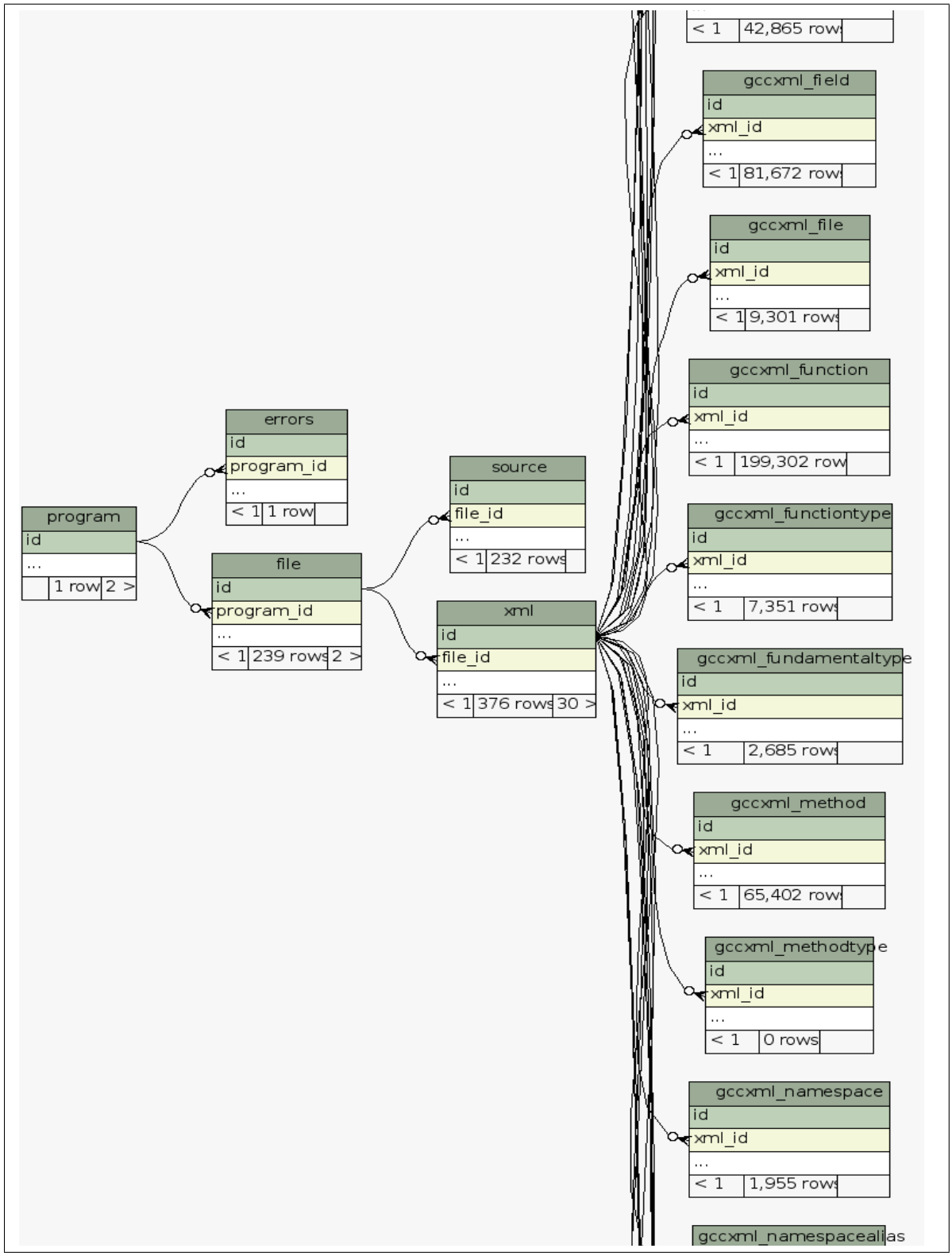

Figure 2: Table Listing of Trike Database Schema

```
MariaDB [trike]> show tables;
+------------------------+
| Tables_in_trike        |
+------------------------+
| errors                 |
| file                   |
| gccxml                 |
| gccxml_argument        |
| gccxml_arraytype       |
| gccxml_base            |
| gccxml_class           |
| gccxml_constructor     |
| gccxml_converter       |
| gccxml_cvqualifiedtype |
| gccxml_destructor      |
| gccxml_enumeration     |
| gccxml_enumvalue       |
| gccxml_field           |
| gccxml_file            |
| gccxml_function        |
| gccxml_functiontype    |
| gccxml_fundamentaltype |
| gccxml_method          |
| gccxml_methodtype      |
| gccxml_namespace       |
| gccxml_namespacealias  |
| gccxml_offsettype      |
| gccxml_operatorfunction|
| gccxml_operatormethod  |
| gccxml_pointertype     |
| gccxml_referencetype   |
| gccxml_struct          |
| gccxml_typedef         |
| gccxml_unimplemented   |
| gccxml_union           |
| gccxml_variable        |
| program                |
| source                 |
| xml                    |
+------------------------+
35 rows in set (0.00 sec)
```

3.3.1 Mapping XML to SQL DDL Statements

GCC-XML generates non-empty XML elements containing other empty and/or non-empty elements containing an arbitrarily complex nesting of other elements. This layout requires an element type to table row mapping with a column for each attribute of the element. This element to table row and attribute to column mapping captures the structural aspects of the code base which enables foreign key references to be used to create implicit relationships between pertinent elements. The type of data each attribute contains dictates the type for the corresponding column in the table for the element. For example, if the attribute has a limited set of valid values such as a 0 or a 1, or of "public," "private," or "protected," an enum was used. If the attribute values are of an unpredictable length and comprised of numbers and characters then text is used. If the attribute values are of predictable length and comprised of numbers and characters then varchar with an appropriate length is used. I acknowledge such an approach might seem arbitrary but in practice it works quite well. A more exhaustive approach to mapping types between SQL and XML is explored in the literature [17, 18, 31, 47, 50]. The minor tradeoff with my naïve approach of mapping XML elements to SQL Data Definition Language (DDL) statements is all entities are treated as sequences of characters. No special considerations are made to treat characters as character data CDATA or parsed character data PCDATA.

CHAPTER IV

Static Analysis with Trike

Trike is a different Static Analysis tool. Trike leverages the power of a relational database to structure code relationships and leverages SQL to provide developers with a method to create their own, unique data extraction queries. The usage of SQL alleviates the burden of learning a restricting tool specific language. Although the ability to write extensions for a Static Analysis tool is not novel, writing such queries in a ubiquitous language like SQL, is to my knowledge, a new approach. To illustrate the usefulness of Trike and the power of using a relational database and SQL, I chose three C++ projects and wrote structurual queries to learn about each project. I then wrote composite queries, using the results of the structural queries to learn even more about each project, and used the knowledge I gained by developing these initial queries to investigate how to begin to write more advanced queries that might help identify errors in these projects and other C++ code bases. The queries I present and discuss in this chapter should prove useful to other developers that want to learn more about these projects as well as their own code. I conclude this chapter with a discussion of queries I would like to see Trike perform but currently cannot due to various limitations imposed upon Trike by GCC-XML 4.3.

4.1 C++ Projects Analyzed

Analyzing the code of an established project helps a developer learn about code design and structure before attempting to introduce changes. Reading source code is one way to analyze code and can lead to a concrete understanding of architecture and implementation. Meta-data can enhance a developers understanding of source code by providing context. In particular, meta-data can help a developer understand the style and preferences of existing developers without having to analyze large code blocks on their own. Preferences like the ratio of public, private and protected data members for classes in the code base might indicate how disciplined developers are at data encapsulation while others like a small average function length might indicate a proclivity to create well defined, reusable, purpose driven pieces of code; large functions might indicate developer immaturity, code obfuscation and a poorly designed code base. Some of these meta-data can be gathered through reading while others cannot. Regardless, it is often easier to have a tool extract it for you, which Trike does quite readily.

Other structural data like the number of Source Lines of Code (SLOC) for the entire project, SLOC per file, average SLOC per class, the number of header files, the number of source files and the number of library files help reveal other important architectural choices. I extract these meta-data and a few others using Trike for the libodc++ library, the Apache log4cxx library and the C++ sockets library and discuss their relevance in the "Existing Trike Queries" 4.2 section. This section is divided into three sub-sections. The first 4.2.2 sub-section describes structural queries 4.2.1, the second describes composite queries derived from structural meta-data 4.2.2 and the third describes progress on developing advanced queries that could be used as the basis for detecting errors 4.2.3. After I introduce a query I describe the thought process behind the development of the query. Initial results seem promising and are provided at the end of the first two sub-sections and full results are presented in the appendices 5.2. Before I present these queries and the initial ouput it will prove helpful to discuss the importance of the projects I chose and elaborate on why I chose

to use them as examples.

The libodc++ Library

Open Database Connectivity (ODBC) software provides software developers with a standardized Application Programming Interface (API) for accessing information in a relational database []. The libodbc++ project uses the API provided by ODBC to build a useful database access library for C++ developers. The project has the single goal of "creat[ing] a complete c++ development environment for SQL database access in [the] form of a class library and a set of tools." The website for the project expands upon this definition and states the following:

> libodbc++ is a c++ class library for accessing SQL databases. It is designed with standards in mind, so it provides a subset of the well-known JDBC 2.0(tm) and runs on top of ODBC. It is distributed under the LGPL. The library has shown to compile and work on Linux, FreeBSD and win32 platforms. Any environment with a reasonable compiler, standard c++ library and ODBC Driver Manager should do. In general, the library should run on anything that has a reasonable c++ environment and an ODBC driver manager version 2.5 or higher.

Developers that use libodbc++ in their C++ projects use it to retrieve data from a relational database. As the core of a programs database logic, libodc++ is one of the most important parts of a C++ program. This is why I chose to analyze the libodc++ project with Trike.

The Apache log4cxx Library

Logging application data is important. Log data can provide insights into user interaction, security, debugging, and other important information. The Apache Software Foundation (ASF) describes the Apache log4cxx library "[a]s a logging

framework for C++ patterned after Apache log4j." Reserved for applications utilizing the Apache Portable Runtime (APR) the Apache log4cxx project is useful for many projects. It is a liberally licensed project, licensed under the Apache License, "is designed to be reliable, fast and extensible ..." and "... strives to be simple to understand and [easy] to use." I chose to analyze log4cxx with Trike for two reasons: 1) log4cxx is used by a large number of C++ projects, including the Apache Web Server and 2) logging is vital to helping developers discover security issues, especially issues related to unanticipated usage scenarios.

The C++ Sockets Library

Sockets are essential to building network enabled applications. The C++ Sockets Library "is a GPL licensed C++ class library wrapping the Berkeley sockets C API, and therefore works on most unices and also win32. The library is in use in a number of real world applications, both commercial and open source." The C++ Sockets Library provides support for the Secure Sockets Layer (SSL), addressing network connected systems using Internet Protocol version 4 (IPv4) and version 6 (IPv6), constructing TCP, UDP and SCTP sockets, and communicating using the HTTP(S) protocol. The library also provides a highly customizable error handling framework and claims to work on Linux and most versions of Windows. It also claims to have partial support for Solaris and Mac OS X. I chose to evaluate the C++ Sockets Library with Trike because it provides C++ developers with an essential piece of code for accessing the network and claims to work on a variety of platforms.

4.2 Existing Trike Queries

SQL queries used to build relationships for data stored in Trike are similar to conventional SQL queries run against data contained in a relational database. Each row in a table, has a column containing a unique key, that distinguishes it from other data in the table. These keys can be used to build relationships with rows contained

in other tables. Some tables contain multiple columns that enable this type of relationship building. Because Trike loads data for each discovered element into the rows of a table, this type of relationship building provides the basis for modeling element interactions in a code base.

I first investigated writing structural queries to learn more about the composition of libodbc++, log4cxx and the C++ Sockets Library. I used the common UNIX utility *wc* to verify the results of some of the queries I developed and when appropriate indicated if the output matched the results in the output tables 4.2.1 4.2.2. I used a hyphen (-) to indicated a query did not have a *wc* equivalent. These checks were important, because *wc* is an established programming tool relied upon many developers to quickly gather structural data on files. They were also important because *wc* is restricted to UNIX and UNIX-like platforms while Trike enables developers on many different platforms to gather the same structural data. Trike is intrinsically cross-platform and supported on the same platforms as MariaDB which include Linux, Solaris, Windows and Mac OS X.

4.2.1 Structural Queries

The first set of queries I developed using Trike were structural queries. The output for most of these queries is contained in a table at the end of this sub-section 4.2.1 and the data which could not be contained in the table is located in the appropriate appendix A-1. When possible and appropriate I indicated in the output table whether the output gathered using *wc* matched the output gathered using my query. In rare instances the data were slightly different. I attributed these discrepancies to documented compilation issues which were captured in the **errors** table or to the duplicate data generated by GCC-XML and subsequently inserted into the database. I attempted to minimize the impact of the latter by utilizing the **distinct()** operator and joins when possible.

When I started developing structural queries for Trike I started at what I believed to be the most logical point, determining how many files were in a project. The SQL for this query is shown in the following block.

The Number of Files in a Program

```
MariaDB [trike]> select count(id) as "Number of Files"
                     from file
                   where type = "source";
```

When I developed this query I utilized the **count()** operator to determine the number of file entities contained in the **file** table. The **file** table serves as the registry for all files in the project. Each entity contained in the **file** table has a *type* which can be "source" or "xml," a *program_id* which corresponds to the unique project identifier for the entity contained in the **program** table, a *name* which contains the short name of the file and a *path* which contains the fully qualified path to the file on the fileystem. The *primary key* for each entity contained in the **file** table is used to build relationships with the generated XML table named **xml** and a few other tables.

After I developed a query to determine the number of files in the project, I then thought about existing Static Analysis tools and realized that the next structural aspect developers would find useful would be determining where code tends to gravitate. The answer to this question involves several queries. Each query helps determine if the majority of code is contained in header files, standard source files or system/library includes. To start to answer this question I first developed a SQL query to determine the number of header files. The SQL query used to determine this is shown in the following block.

The Number of Header Files in a Program

```
MariaDB [trike]> select count(id) as "Number of Header Files"
                     from file
                   where type = "source"
                     and name like "%.h";
```

This query also uses the **file** table and the **count()** operator. I made it more specific by selecting the file type as well as the file extension. In this case the appropriate file extensions were *.h* or *.hpp*. I used *.h* because the projects I analyzed did not contain *.hpp* files.

The next SQL query I wrote helped determine the number of source files present in the code base. It is very similar to the previous query, with the one exception of file

extension. Instead of using *.h* or *.hpp* I used *.cpp*. The SQL query used to determine this number is shown in the following block.

The Number of Source Files in a Program

```
MariaDB [trike]> select count(id) as "Number of Source Files"
                     from file
                 where type = "source"
                     and name like "%.cpp";
```

I then though about code locality beyond file counts and wondered what kind of query could be constructed to determine the Source Lines of Code (SLOC) for the entire program, and then for each file in the program. The SQL for the determining the SLOC for the entire program is shown in the following block.

The Total Lines in a Program (SLOC)

```
MariaDB [trike]> select
                     sum(
                         (
                          length(source.contents)
                          -
                          length(replace(source.contents,"\n",""))
                         )
                        )
                        as "SLOC"
                 from source,file
                     where source.file_id = file.id;
```

The SLOC query uses the **source** table which contains a *file_id* column referencing the appropriate entity from the **file** table and a *contents* column used to store the entire contents of the referenced file. The SLOC calculation is performed in the database using the **sum()** and **length()** operators.

I then thought about the SLOC for a program in terms of my original question about determining the number of files in the project. This led me to try to determine how many lines of code were in each individual file. The SQL query for determining the SLOC for each file is shown in the following block. The results for this query are included in the appendix A-1.

The SLOC per File

```
MariaDB [trike]> select file.id,file.name,
                      length(source.contents)
                      -
                      length(replace(source.contents,"\n",""))
                          as "SLOC"
                  from file,source
                      where file.type = "source"
                          and file.id = source.file_id;
```

I then thought about how these numbers could be used to tell me how many lines of code were in header files and how many lines of code were in source files. These data would complement the previous queries for determining the number of header files and source files by creating a clearer picture of code locality. The first SQL query, the one for determining the SLOC in header files is shown in the following block. The results for this query are included in the appendix A-1.

The SLOC for all Header Files

```
MariaDB [trike]> select
                      sum(
                          length(source.contents)
                          -
                          length(replace(source.contents,"\n",""))
                         )
                      as "The SLOC in Header Files"
                  from file,source
                      where file.type = "source"
                          and file.id = source.file_id
                          and file.name like "%.h";
```

This query is similar to the query for determing the SLOC for a program but excludes files which are not header files by filtering files based on their extension. In this case I used *.h* to create a filter for header files.

The second SQL query, the one for determining the SLOC in source files is shown in the following block and is quite similar to the one used to determine SLOC for each file. Instead of using *.h* or *.hpp* I used *.cpp* to select source files. The results for this query are included in the appendix A-1.

The SLOC for all Source Files

```
MariaDB [trike]> select
                      sum(
                          length(source.contents)
                          -
                          length(replace(source.contents,"\n",""))
                         )
                         as "The SLOC in Source Files"
                  from file,source
                      where file.type = "source"
                          and file.id = source.file_id
                          and file.name like "%.cpp";
```

I then wanted to know how many bytes were contained in each of the files to help estimate source size. This query does not introduce anything that has not already been utilized in other queries but instead of generating a difference between file contents with newlines characters and without newlines characters we simply count the bytes in the source file. The SQL query for the byte count is shown in the following block and the results are included in the appendix A-1.

The Bytes per File

```
MariaDB [trike]> select file.name,length(source.contents)
                      as "Byte Count"
                  from source,file
                      where source.file_id = file.id;
```

As an artifact of the number of bytes used to store characters on my platform, the byte count also corresponded to the number of characters contained in each file. This would not be true for locales requiring more than one byte to represent a character but could be easily adjusted to accommodate different locales if a developer wanted to use this query to calculate the number of characters in a file written in their native language.

After determining the size of files in the code base I wanted to improve my understanding of file contents and code locality. The first query I explored was calculating the number of functions in the entire code base including system and library files. The SQL query for this calculation is shown in the following block.

The Number of Functions in a Program

```
MariaDB [trike]> select count(distinct(name))
                     as "Function Count"
                 from gccxml_function;
```

This is the first query to utilize the **distinct()** function to eliminate duplicates generated by GCC-XML and inserted into the GCC-XML function table named **gccxml_function**. I then realized that it would be helpful to determine the number of functions contained in each generated XML file. The SQL query for this calculation is shown in the following block and is similar to the previous query but I wanted to see if I could elicit a per file function count using the **gccxml_function** table and the **gccxml_file** table. It proved difficult to use the raw source for this calculation, due to the structure of the data generated by GCC-XML, but it was possible using the generated xml file contents. The results for this query are included in the appendix A-1.

The Number of Functions per Generated XML File

```
MariaDB [trike]> select gccxml_file.xml_id,count(distinct(gccxml_function.id))
                     as "Function Count per XML File"
                 from gccxml_function,gccxml_file
                     where gccxml_function.xml_id = gccxml_file.xml_id
                         group by gccxml_file.xml_id;
```

I then wanted to count the number of distinct classes in the code base. The SQL query for calculating this number is shown in the following block. This query is very similar to the function count query only instead of using the **gccxml_function** table it uses the **gccxml_class** table.

Total Program Class Count

```
MariaDB [trike]> select count(distinct(name))
                     as "Class Count"
                 from gccxml_class;
```

After calculating the number of functions, the number of functions per file, and the number of classes in each program, I then wanted to count the number of distinct variables in the code base as this could help me prepare tests for evaluating

the code. The more variables that exist in the code base, the more states I would have to reach and the more tests I would have to write to thoroughly evaluate the code base. The query I developed utilizes methods introduced in previous queries but used the **gccxml_variable** table. The SQL query for calculating this number is shown in the following block.

Total Variable Count

```
MariaDB [trike]> select count(distinct(name))
                     as "Variable Count"
                 from gccxml_variable;
```

After calculating the total number of variables in the code base I then thought it would be useful to divide the number into private, protected and public class variables contained in the code base. The query I developed use the same approach as the "Total Variable Count" query but utilizes the **access** column for entities stored in the **gccxml_variable** table. Valid values for the "access" column include "private," "protected," or "public." Determining the number of each of these variables should help me learn more about how well or how poorly data is encapsulated throughout the code base as well as help with testing. The SQL query to calculate the number of private variables is shown in the following block and can easily be changed to calculate protected and public variables by adjusting the value of the "access" parameter as shown in the two subsequent queries.

Total Private Variable Count

```
MariaDB [trike]> select count(distinct(name))
                     as "Private Variable Count"
                 from gccxml_variable
                     where access = "private";
```

Total Protected Variable Count

```
MariaDB [trike]> select count(distinct(name))
                     as "Protected Variable Count"
                 from gccxml_variable
                     where access = "protected";
```

Total Public Variable Count

```
MariaDB [trike]> select count(distinct(name))
                     as "Public Variable Count"
                 from gccxml_variable
                     where access = "public";
```

Now that I knew more about each project I wanted to start learning more about the code each of these projects depended on. The first question I had was what system level and library files were included by GCC while generating the intermediate representation. The SQL query for generating this list of dependencies is shown in the following block.

List of Included Library Files

```
MariaDB [trike]> select distinct(gccxml_file.name)
                     from gccxml_file, xml
                 where xml.id in
                     (
                      select distinct(gccxml_file.xml_id)
                          from gccxml_file
                     )
                     and gccxml_file.fid not in
                         (select id from file);
```

This query uses previous methods, like **distinct()** to remove duplicate data, but also introduces the concept of sub-selects where I generate a list of files and use those lists to filter potential candidates from the final result set. The results of this query are included in the appendix A-1.

I then wanted to calculate the number of system level and library dependencies included by GCC. This should help later when I seek to determine the percentage of code reuse attributable to code contained outside of the code base versus code contained inside of the code base. The SQL query for determining this number is shown in the following block. It uses methods and tables introduced through previous queries.

The Number of Included Library Files

```
MariaDB [trike]> select count(distinct(gccxml_file.name))
                     as "Number of Included Library Files"
                 from gccxml_file, xml
                     where xml.id in
                         (select distinct gccxml_file.xml_id from gccxml_file)
                             and gccxml_file.fid not in
                                 (select id from file);
```

I then wanted to determine the particular importance of a file in the source base. The level of importance is determined by calculating the number of times a particular file is referenced in other files as an include. The SQL query for determining these numbers for all files in the project code base is shown below and the results are shown in the appendix A-1.

File Weight

```
MariaDB [trike]> select file.id,file.name,
                         (select count(gccxml_file.id)
                              from gccxml_file
                          where file.id = gccxml_file.fid)
                     as "File Weight"
                 from file
                     where file.type = "source";
```

The results from this query revealed something interesting: many files had a zero file weight. Further investigation warranted, I looked at the source for a small sample of files and verified that indeed the files with a zero file weight were not included in any other source files in the project code base. Some were included in tests but test directories were excluded from the discovery process. This data is priceless when a developer wants to determine if they can safely remove a file from code and in situations when they need to determine if they need to test the code contained in a file or are trying to localize an issue and need to know if they should check particular files or not. These results might also indicate the code is simply standalone code. In either case it still gives the developer context when analyzing the code base.

If a developer wants to determine the file weight for a particular file they first

need to determine the unique id for the file in the **file** table. The SQL query to determine the id for the "Socket.cpp" file in the C++ Sockets Library is shown below along with the output.

Get the File ID for a Particular File

```
MariaDB [trike]> select id
                     as "FID"
                 from file
                     where name = "Socket.cpp"
                         and type = "source";
```

The File ID for the Socket.cpp file in the C++ Sockets Library

```
+-----+
| FID |
+-----+
| 114 |
+-----+
1 row in set (0.00 sec)
```

After the developer determines this number they then substitute this number for the text marked "FID" in the SQL query shown below to determine the file weight.

Get the File Weight for a Particular File

```
MariaDB [trike]> select count(gccxml_file.id)
                     as "File Weight"
                 from gccxml_file
                     where fid = "FID";
```

There were many queries discussed in this sub-section. I first started with a query to determine the number of files contained in a program. I then asked about code locality, where code is written and new functionality is typically added. These questions were answered in part by developing queries to tell me how many header files, source files, and system and library files were in a program. I then started to ask more questions about the code base such as how many functions, classes and variables existed in the code base. Like previous questions, initial queries were developed to begin to answer these questions. Finally, I wrote queries to help me develop an understanding of how data is (or is not) encapsulated in the project and what code is

Query	*libodbc++*	*log4cxx*	*sockets*	*wc matches?*
The Number of Files	42	325	122	-
The Number of Header Files	23	177	67	-
The Number of Source Files	19	148	55	-
Total Source Lines of Code (SLOC)	16444	45733	26869	Y
Total Header SLOC	6408	22356	8310	Y
Total Source SLOC	10036	23377	18559	Y
The Number of Functions	1298	1680	3331	-
Total Class Count	361	752	280	-
Total Variable Count	392	1050	285	-
Total Private Variable Count	81	23	40	-
Total Protected Variable Count	0	0	1	-
Total Public Variable Count	311	1029	244	-
Number of Included Files	170	159	139	-

Table I: Structural Query Output

actually used in the project. Each query added to my understanding of the code base for each project by providing a distinct piece of meta-data as shown in the table at the end of this sub-section. Each piece of meta-data proved useful on its own and as I will soon show will be made even more useful when combined with other meta-data 4.2.2.

4.2.2 Composite Queries

After developing the structural queries in the previous section 4.2.1, I began to think about how the results could be combined to build a better picture of the code base. The first question I asked combined the SLOC for each file and the number of files to derive the average SLOC per file. This number is important to determine the average SLOC of the files contained in the code base. A large number of files coupled with a low average SLOC or a small number of files coupled with a high average SLOC might indicate poor design. The SQL query for this calculation is shown below and utilizes "The Total Lines in a Program (SLOC)" query and "The Number of Source Files in a Program" query.

Average SLOC for All Files

```
MariaDB [trike]> select
                      sum(
                          (
                           length(source.contents)
                           -
                           length(replace(source.contents,"\n",""))
                          )
                         )
                      /
                      count(file.id)
                          as "Average SLOC"
                  from file,source
                      where file.type = "source"
                          and file.id = source.file_id;
```

I then asked the same question for just header files. The SQL query for this calculation is shown below and uses the "The SLOC for all Header Files" query and "The Number of Header Files in a Program" query.

Average SLOC for Header Files

```
MariaDB [trike]> select sum(
                             (
                              length(source.contents)
                              -
                              length(replace(source.contents,"\n",""))
                             )
                            )
                         /
                         count(file.id)
                             as "Average .h SLOC"
                  from file,source
                      where file.type = "source"
                           and file.id = source.file_id
                           and file.name like "%.h";
```

I then asked the same question for just source files. The SQL query for this calculation is shown below and uses the "The SLOC for all Source Files" query and "The Number of Source Files in a Program" query.

Average SLOC for Source Files

```
MariaDB [trike]> select sum(
                              (
                               length(source.contents)
                               -
                               length(replace(source.contents,"\n",""))
                              )
                             )
                           /
                           count(file.id)
                               as "Average .cpp SLOC"
                  from file,source
                      where file.type = "source"
                          and file.id = source.file_id
                          and file.name like "%.cpp";
```

After calculating the average SLOC for the entire program and then for header and source files, I then returned to a question I raised in the previous section regarding the density of variables with particular access 4.2. To calculate the proportion of private, protected and public variables in the code base I combined the queries I used to calculate them individually into one query using the **concat()** operator. The query for calculating this proportion is shown below.

Proportion of Private, Protected and Public Variables

```
MariaDB [trike]> select
                       concat(
                           (select count(distinct(name))
                                from gccxml_variable
                            where access = private),
                            to ,
                           (select count(distinct(name))
                                from gccxml_variable
                            where access = protected),
                            to ,
                           (select count(distinct(name))
                                from gccxml_variable
                            where access = public)
                       )
                  as Private to Protected to Public;
```

We can also calculate the percentage of the total variables for each level of visibility compared to the total number of variables in the program. This percentage

is another representation of the data the proportion data reveals and can also help estimate how well data might be encapsulated throughout the code base. The SQL query for calculating the percentage of "private" variables is shown below with a subsequent queries for "protected" and "public" variables.

Percentage Private Variables

```
MariaDB [trike]> select
                      concat(
                          ((select count(distinct(name))
                                from gccxml_variable
                            where access = "private")
                          /
                          (select count(distinct(name))
                               from gccxml_variable))
                          * 100,)as "Percentage Private Variables";
```

Percentage Protected Variables

```
MariaDB [trike]> select
                      concat(
                          ((select count(distinct(name))
                                from gccxml_variable
                            where access = "protected")
                          /
                          (select count(distinct(name))
                               from gccxml_variable))
                          * 100,)as "Percentage Protected Variables";
```

Percentage Public Variables

```
MariaDB [trike]> select
                      concat(
                          ((select count(distinct(name))
                                from gccxml_variable
                            where access = "public")
                          /
                          (select count(distinct(name))
                               from gccxml_variable))
                          * 100,)as "Percentage Public Variables";
```

After I explored the visibility of variables in the program I then investigated developing a query to try and determine which percentage of the code base was actually being used to generate the program. This idea came about after I made the

discovery in the previous section 4.2 that some files were not referenced by any other file. This query proved elusive and in order to calculate the actual percentage of code being used in the code base I needed to develop a set of three queries. The first query returned all files that are referenced as includes. The second query retrieved the number of rows in the first result and the third query calculated the percentage of files referenced as includes over the total number of files in the project. The set of SQL queries are shown below and the results are included in the table.

Set of Queries to Calculate Percentage of Files Referenced

```
-- QUERY #1
MariaDB [trike]> select (
                     select count(gccxml_file.id)
                         from gccxml_file
                     where file.id = gccxml_file.fid
                 )
                     as "File_Weight"
                 from file
                     where file.type = "source"
                         group by file.id
                             HAVING File_Weight > 0;

-- QUERY #2
MariaDB [trike]> select found_rows()
                     as "Number Files With Dependencies";

-- QUERY #3
MariaDB [trike]> select concat(
                         ("QUERY #2 OUTPUT"/"TOTAL FILES") * 100
                        ,"%")
                         as "Percentage of Files Referenced"
                 from dual;
```

The final query I developed was to calculate the percentage of code reuse measured by the number of system and library included from outside of the native code base over the total number of files used to build the final executable. The SQL query to calculate this number is shown below and uses a variety of concepts and

Query	*libodbc++*	*log4cxx*	*sockets*	*wc matches?*
Average SLOC	391.5238	140.7169	220.2377	Y
Average Header SLOC	278.6087	126.3051	124.0299	Y
Average Source SLOC	528.2105	157.9527	337.4364	Y
Percentage Private Variables	20.6633	2.1905	14.0351	-
Percentage Protected Variables	0.0000	0.0000	0.3509	-
Percentage Public Variables	79.3367	98.0000	85.6140	-
Percentage Files Referenced	71.4286	97.8462	68.0328	-

Table II: Composite Query Output

tables introduced through previous queries. The results are shown in the appendix A-2.

Percent Code Reused

```
MariaDB [trike]> select
                      concat(
                             (
                              (select count(distinct(gccxml_file.name))
                                   from gccxml_file,xml
                               where xml.id in
                                   (select distinct(gccxml_file.xml_id)
                                        from gccxml_file)
                               and gccxml_file.fid not in (select id from file)
                              )
                              /
                              (select count(id)
                                   from file
                               where type = "source")
                             ),"%"
                            )
                               as "Percent Code Reused"
                  from dual;
```

This sub-section used some of the queries and meta-data developed in the structural query sub-section to extract more complex attributes. These meta-data are shown in the table included in this sub-section and in the appendix. The list of composite queries presented in this sub-section is by no means exhaustive but the ones I explored provided a glimpse into the many meaningful relationships that can be constructed using meta-data extracted by Trike. In the next section I discuss a few advanced queries that should provide a foundation for when I start work on

modifying Trike to identify errors in a program.

4.2.3 Advanced Queries

Now that I have established the utility of a relational database with regard to structural static program analysis, it makes perfect sense to discuss how Trike might be used in the future to identify errors. In addition to the structural and composite queries discussed in the previous sub-sections 4.2.14.2.2, the meta-data generated by Trike should enable developers to start exploring deeper relationships in their code base. The first step in this metamorphosis is to build context. Most errors detected by static analysis tools deal with order to provide context. In order to generate pre and post-conditions Trike needs to extract meaningful data before important markers like function calls and meaningful data after each marker. As it currently stands Trike is limited by the underlying power of GCC-XML. Nonetheless, I still managed to determine what a developer might be able to do to start detecting errors in their code base. The output for some of these queries is included in the appendix A-3.

The first step would be to identify a useful snippet of code to investigate. I have chosen a class member named "Data" in the Event class located in the C++ Sockets Library. Once this snippet of code is isolated the developer can then use variations of the following queries to extract data before and after the snippet of interest. This approach is by no means complete, but will be once Trike is able to import pertinent code as an element attribute. The queries required for this approach are shown below and for completeness, I have shown these queries using the raw source and the generated XML.

Search Source for a String

```
1 MariaDB [trike]> select file.name,source.id,source.file_id,
2                          locate(Event::Data() const,source.contents)
3                              as located_at_char
4                   from source,file
5                       where source.file_id=file.id;
```

Search XML for a String

```
MariaDB [trike]> select file.name,xml.id,xml.file_id,
                     locate(Event::Data() const,xml.contents)
                         as located_at_char
                 from xml,file
                     where xml.file_id=file.id;
```

Find Code Surrounding a String in Source

```
MariaDB [trike]> select file.name,source.id,source.file_id,
                  substr(source.contents from
                              locate(Event::Data() const,source.contents) - 20 for 40)
                       as data
              from source, file
                  where source.file_id=file.id
                      limit 1;
```

Find XML Surrounding a String in XML

```
MariaDB [trike]> select file.name,xml.id,xml.file_id,
                     substr(xml.contents from
                                 locate(Event::Data() const,xml.contents) - 20 for 40)
                         as data
                 from xml,file
                     where xml.file_id=file.id
                         limit 1;
```

Each of these queries give particular insight into the developers code base and form the basis for contextual analysis, as well as helping a developer begin to identify the various classes of errors outlined in the "Introduction" 1. I believe that with changes to GCC-XML or by using a more extensive tool like CIL, contextual analysis would become a reality in Trike and enable developers to identify errors in their code.

4.3 Further Query Discussion

In this section I introduced you to several queries you can use on your own C++ projects to extract useful meta-data. The first sub-section discussed various structural queries. Although many of the queries revealed useful meta-data, the initial set of queries I proposed could be improved. In particular, it would have been nice to

give developers a better understanding of functions and classes with respect to the source files and not the generated XML but the the data generated by GCC-XML proved limiting. It would also have been nice to not have to filter duplicates with the "distinct" function but the approach used to generate the XML with GCC-XML again proved limiting. Regardless of these meta-data should provide insight into each code base. In the second sub-section, I combined queries in the first sub-section to derive compound relationships. In addition to the average SLOC it would have been interesting to determine the median and mode for files in each code base as these values would provide additional insight into code placement. It was surprising to observe the over-reliance on public class members by each project and a study attempting to correlate security issues related to this reliance might prove interesting. It was also interesting to see the amount of code reused by each program. These numbers were not surprising as part of the power developers have nowadays is given to them through code reuse. Code reuse can reduce vulnerabilities and speed development so it is not surprising to see system libraries used so heavily. Libraries are also often required for code to work on a specific operating system, which could also be another reason why so many external library files were listed. Finally, in the third sub-section I discussed a few queries that should be vital to the future development of Trike as a competitive static analysis tool. Trike is currently very useful for performing structural static analysis but needs more work to identify errors in C++ code. Nonetheless, the results of these queries indicate the relational database should be regarded as a capable tool for performing static program analysis.

CHAPTER V

Discussion

Trike is useful for generating structural meta-data about C++ programs that can be built by GCC. Structural static analysis is useful in its own right as it enables developers to build an understanding of their code base from a compositional standpoint while enabling evolutionary structural analysis. The work I presented in this thesis represents the beginning of what one can do with a database as a Static Analysis tool. In order to access more complex aspects of Static Analysis, GCC-XML needs to be improved to expose and import particular data about the source code such as the code specific to functions and classes. It is not enough to attempt to build references to code using the raw source and elements from the database using the current version of GCC-XML.

Another area that I will have to investigate further is the grammatical completeness of GCC-XML. GCC-XML supports many important C++ elements but some might have been omitted. This is an acknowledged shortcoming of GCC-XML. Some of the work to rectify this problem is started and mentioned on the GCC-XML mailing list but much of it has yet to be patched into the mainline of the code base. Once these modifications are made, I will have to make changes to the core Trike logic to accommodate them but I am not certain at this point where such changes might be made beyond modifications to the DDL statements, parsing of the XML, and un-

derlying SQL statements to insert the parsed data. The possibility also remains that GCC-XML might need to be replaced by a different tool like CIL.

5.1 Potential Limitations

Trike is currently limited by the completeness of GCC-XML and my current list of queries. Like many other Static Analysis tools Trike is limited in scope both by language and by the type of errors it can detect. It is conceivable that some of these limitations can easily be overcome by extensions such as small programs performing operations outside of the database, stored procedures and functions performing operations inside of the database, as well as improvements to the underlying components like GCC-XML which often forced me to use functions like **distinct( )** to derive useful data from the stored XML.

5.2 Areas for Improvement

After the changes discussed in 5 are made I plan to make the interface to Trike even better. SQL is a useful tool for gathering data but not all developers have a complete grasp of the various standards and subtle nuances of particular implementations like those found in MariaDB. In order to make regular routines more useful and to enable the development of external code frameworks I am currently planning for Trike to include stored procedures and functions to make particular types of analysis programmatically accessible. Once this is accomplished I plan to create simple libraries for popular dynamic languages like Perl, PHP and Python to expose the power of Trike for users to include in their own work-flows.

Trike is a useful tool and most importantly an approachable tool. However, it is new and requires more work. The generated XML is potentially incomplete when compared to the various C++ specifications. The current DTD is useful but as the generated XML is further inspected, changes to the internals of GCC-XML will invariably result in modifications to the existing DTD. The database currently sup-

ports one project and unless the developer is exploring the *Partial Program Analysis* aspects of Trike, it is regenerated for every run. The final area that needs further exploration is the area of *Partial Program Analysis*. Trike is able to regenerate XML on a per file basis but how this regeneration cleanly fits within the entire database model requires further research before a concise replacement of specific elements can be guaranteed.

BIBLIOGRAPHY

[1] Apple Computer. Xcode: Integrated development environment. `http://developer.apple.com/technologies/mac/#xcode`, July 2011.

[2] A. Beaulieu. *Learning SQL*. O'Reilly, 1st edition, 2005.

[3] B. Boehm. A view of 20th and 21st century software engineering. In *Proceedings of the 28th international conference on Software engineering*, ICSE '06, pages 12–29, New York, NY, USA, 2006. ACM.

[4] B. W. Boehm. *Software Engineering Economics*. Prentice Hall, 2nd edition, 1981.

[5] C. Cadar, D. Dunbar, and D. Engler. Klee: Unassisted and automatic generation of high-coverage tests for complex systems programs. Technical report, Stanford, Stanford University, 353 Serra Mall, Stanford, California 94305, 2008.

[6] S. S. Chawathe. Describing and manipulating xml data. *Special Issue on XML, IEEE Computer Society*, 22(3):3–9, 1999.

[7] H. Chen and D. Wagner. MOPS: an Infrastructure for Examining Security Properties of Software. In *Proceeding of the 9th ACM conference on Computer and communications security*, CCS '02, New York, NY, USA, 2002. ACM.

[8] C. N. Christopher. Evaluating Static Analysis Frameworks. Technical report, Carnegie Mellon University, May 2006.

[9] E. Codd. A relational model of data for large shared data banks. *Information Retrieval, Communications of the ACM*, 13(6):377–387, 1970.

[10] Cppcheck Developers. Cppcheck. `http://cppcheck.sourceforge.net/`, June 2011.

[11] A. Dasgupta, V. R. Narasayya, and M. Syamala. A static analysis framework for database applications. In *International Conference on Data Engineering*, pages 1403–1414, 2009.

[12] A. Deutsch, M. Fernandez, D. Florescu, A. Levy, D. Maier, and D. Suciu. Querying xml data. *Special Issue on XML, IEEE Computer Society*, 22(3):10–18, 1999.

[13] I. Dillig, T. Dillig, and A. Aiken. SAIL: Static Analysis Intermediate Language with a Two-Level Representation. Technical report, Stanford, 2009.

[14] D. Engler and M. Musuvathi. Static Analysis versus Software Model Checking for Bug Finding. In *In VMCAI*, pages 191–210. Springer, 2004.

[15] D. Evans and D. Larochelle. Improving security using extensible lightweight static analysis. *IEEE Softw.*, 19:42–51, January 2002.

[16] N. Flemming, H. R. Nielson, and C. Hankin. *Principles of Program Analysis*. Springer, 2nd edition, 2005.

[17] D. Florescu and D. Kossmann. A performance evaluation of alternative mapping schemes for storing xml data in a relational database. Technical report, Inria, France and University of Passau, Germany, 1999.

[18] D. Florescu and D. Kossmann. Storing and Querying XML Data Using an RDBMS. *Special Issue on XML, IEEE Computer Society*, 22(3):27–34, 1999.

[19] A. Fokin, K. Troshina, and A. Chernov. Reconstruction of class hierarchies for decompilation of c++ programs. *Software Maintenance and Reengineering, European Conference on*, 0:240–243, 2010.

[20] D. Galin. *Software Quality Assurance*. Pearson, 1st edition, 2004.

[21] GCC-XML Developers. GCC-XML DTD. `http://www.gccxml.org/files/v0.6/gccxml-2004-11-19.dtd.txt`, June 2011.

[22] GCC-XML Developers. GCC-XML Mailing List - DTD Beginnings. `http://www.cmake.org/pipermail/gccxml/2003-July/000237.html`, June 2011.

[23] GCC-XML Developers. GCC-XML Mailing List - DTD Update. `http://www.cmake.org/pipermail/gccxml/2004-November/000511.html`, June 2011.

[24] GCC-XML Developers. GCC-XML: The XML output extension to GCC. `http://www.gccxml.org/HTML/Index.html`, June 2011.

[25] Google Inc. and Cpplint Developers. Cpplint. `http://sourceforge.net/projects/cpplint/`, June 2011.

[26] T. Hofer. Evaluating Static Source Code Analysis Tools. Master's thesis, M-Icole Polytechnique FM-IdM-Irale de Lausanne, Lausanne, Switzerland, 2010.

[27] D. Hovemeyer, J. Spacco, and W. Pugh. Evaluating and tuning a static analysis to find null pointer bugs. *SIGSOFT Softw. Eng. Notes*, 31:13–19, September 2005.

[28] M. F. II, L. Marrs, and B. G. Ryder. Visualizing the Results of a Complex Hybrid Dynamic-Static Analysis. Technical report, Virginia Tech, 114 McBryde Hall (0106) Virginia Tech Blacksburg, VA 24061, 2010.

[29] S. H. Kan. *Metrics and Models in Software Quality Engineering*. Pearson, 1st edition, 2003.

[30] Y. P. Khoo, J. S. Foster, M. Hicks, and V. Sazawal. Path projection for user-centered static analysis tools. In *Proceedings of the 8th ACM SIGPLAN-SIGSOFT workshop on Program analysis for software tools and engineering*, PASTE '08, pages 57–63, New York, NY, USA, 2008. ACM.

[31] R. Krishnamurthy, R. Kaushik, and J. Naughton. XML-to-SQL Query Translation Literature: The State of the Art and Open Problems. In Z. Bellahsène, A. Chaudhri, E. Rahm, M. Rys, and R. Unland, editors, *Database and XML Technologies*, volume 2824 of *Lecture Notes in Computer Science*, pages 1–18. Springer Berlin / Heidelberg, 2003. 10.1007/978-3-540-39429-7_1.

[32] B. Livshits. *Improving Software Security with Precise Static and Runtime Analysis*. PhD thesis, Stanford University, Palo Alto, CA, USA, 2006.

[33] LLVM Developers. The Clang Static Analyzer. `http://clang-analyzer.llvm.org/`, June 2011.

[34] M. O. McCracken. The Design and Implementation of the LENS Program Information Framework. Technical report, UCSD, 2006.

[35] J. Midtgaard. Control-flow Analysis of Functional Programs. Technical report, University of Aarhus, IRISA/INRIA, BRICS, Department of Computer Science, University of Aarhus, IT-parken, Aabogade 34, DK-8200, Aarhus, N, Denmark, November 2007.

[36] A. Mockus and D. M. Weiss. Predicting risk of software changes. *Bell Labs Technical Journal*, 5:169–180, 2000.

[37] M. S. Musuvathi, D. Park, D. Y. W. Park, A. Chou, D. R. Engler, and D. L. Dill. Cmc: A pragmatic approach to model checking real code. In *In Proceedings of the Fifth Symposium on Operating Systems Design and Implementation*, 2002.

[38] N. Nagappan and T. Ball. Static analysis tools as early indicators of pre-release defect density. In *Proceedings of the 27th international conference on Software engineering*, ICSE '05, pages 580–586, New York, NY, USA, 2005. ACM.

[39] N. Nagappan, T. Ball, and A. Zeller. Mining Metrics to Predict Component Failures. Technical report, Microsoft Research, 2005.

[40] G. C. Necula, S. Mcpeak, S. P. Rahul, and W. Weimer. CIL: Intermediate Language and Tools for Analysis and Transformation of C Programs. In *In International Conference on Compiler Construction*, pages 213–228, 2002.

[41] Oracle, Inc. MySQL. `http://dev.mysql.com`, June 2011.

[42] D. Quinlan and C. Cohen. Discovery of C++ Data Structures from Binaries. *Cyber Security and Information Intelligence Research Workshop, ACM 6th Annual*, 0:1–3, 2010.

[43] N. Rutar, C. B. Almazan, and J. S. Foster. A Comparison of Bug Finding Tools for Java. In *Proceedings of the 15th International Symposium on Software Reliability Engineering*, pages 245–256, Washington, DC, USA, 2004. IEEE Computer Society.

[44] SchemaSpy Developers. SchemaSpy: Graphical Database Schema Metadata Browser. `http://schemaspy.sourceforge.net/`, July 2011.

[45] B. Schwarz, H. Chen, D. Wagner, J. Lin, W. Tu, G. Morrison, and J. West. Model checking an entire linux distribution for security violations. In *Proceedings of the 21st Annual Computer Security Applications Conference*, pages 13–22, Washington, DC, USA, 2005. IEEE Computer Society.

[46] R. W. Selby. *Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management, and Research*. IEEE Computer Society, 1st edition, 2007.

[47] I. Tatarinov, S. D. Viglas, K. Beyer, J. Shanmugasundaram, E. Shekita, and C. Zhang. Storing and querying ordered xml using a relational database system. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, SIGMOD '02, pages 204–215, New York, NY, USA, 2002. ACM.

[48] The Eclipse Foundation. Eclipse: Integrated Development Environemnt. `http://www.eclipse.org/`, July 2011.

[49] The GNU Project. GCC: The GNU Compiler Collection. `http://gcc.gnu.org`, July 2011.

[50] K. K. —ver Jensen. Type Checking of SQL/XML. Master's thesis, University of Aarhus, Denmark, 2010.

[51] W3C: World-Wide Web Consortium. Extensible Markup Language (XML). http://www.w3.org/XML/, July 2011.

[52] W3C: World-Wide Web Consortium. W3C XML Query (XQuery). http://www.w3.org/XML/Query/, July 2011.

[53] W3C: World-Wide Web Consortium. XML Path Language (XPath). http://www.w3.org/TR/xpath/, July 2011.

[54] W3C: World-Wide Web Consortium. XSL Transformations (XSLT) Version 2.0. http://www.w3.org/TR/xslt20/, July 2011.

[55] M. Widenius. MariaDB. http://mariadb.org, June 2011.

[56] D. Wilkerson, K. Chen, and S. McPeak. Oink: a Collaboration of C/C++ Tools for Static Analysis and Source-to-Source Transformation. http://daniel-wilkerson.appspot.com/oink/index.html, June 2011.

[57] T. Witkowski, N. Blanc, D. Kroening, and G. Weissenbacher. Model checking concurrent Linux device drivers. In *22nd IEEE International Conference on Automated Software Engineering (ASE)*, pages 501–504. IEEE, 2007.

[58] D. Worth, C. Greenough, and L. Chin. A Survey of C and C++ Software Tools for Computational Science. Technical report, Science and Technology Facilities Council, Library and Information Services, SFTC Rutherford Appleton Laboratory, Harwell Science and Innovation Campus, Didcot OX11 0QX, UK, December 2009.

[59] V. Yerramilli. Static Analysis of Novice Student C++ Programs. Master's thesis, Texas Tech University, Texas, USA, 1998.

[60] A. Zeller. Learning from 6,000 projects: Mining models in the large. *Source Code Analysis and Manipulation, IEEE International Workshop on*, 0:3–6, 2010.

[61] H. Zheng, K. Zhou, X. Lai, C. Liu, and Z. Chi. Research on XML Based Static Analysis Software Security Analysis. In *Software Engineering (WCSE), 2010 Second World Congress On*, volume 2, pages 141–144, 2010.

APPENDIX

A-1 Structural Query Output

A-1.1 The Number of Files

The output for the libodbc++ project for this query is

The Number of Files in libodbc++

```
+------------------+
| Number of Files |
+------------------+
|               42 |
+------------------+
1 row in set (0.00 sec)
```

for the log4cxx project the output is

The Number of Files in log4cxx

```
+------------------+
| Number of Files |
+------------------+
|              325 |
+------------------+
1 row in set (0.00 sec)
```

and for the C++ Sockets Library the output is

The Number of Files in the C++ Sockets Library

```
+------------------+
| Number of Files |
+------------------+
|              122 |
+------------------+
1 row in set (0.00 sec)
```

A-1.2 The Number of Header Files

The output for the libodbc++ project for this query is

The Number of Header Files in libodbc++

```
+-------------------------+
| Number of Header Files |
+-------------------------+
|                      23 |
+-------------------------+
1 row in set (0.00 sec)
```

for the log4cxx project the output is

The Number of Header Files in log4cxx

```
+-------------------------+
| Number of Header Files |
+-------------------------+
|                     177 |
+-------------------------+
1 row in set (0.00 sec)
```

and for the C++ Sockets Library the output is

The Number of Header Files in the C++ Sockets Library

```
+-------------------------+
| Number of Header Files |
+-------------------------+
|                      67 |
+-------------------------+
1 row in set (0.00 sec)
```

A-1.3 The Number of Source Files

The output for the libodbc++ project for this query is

The Number of Source Files in libodbc++

```
+------------------------+
| Number of Source Files |
+------------------------+
|                     19 |
+------------------------+
1 row in set (0.00 sec)
```

for the log4cxx project the output is

The Number of Source Files in log4cxx

```
 +------------------------+
| Number of Source Files |
+------------------------+
|                    148 |
+------------------------+
1 row in set (0.00 sec)
```

and for the C++ Sockets Library the output is

The Number of Source Files in the C++ Sockets Library

```
+------------------------+
| Number of Source Files |
+------------------------+
|                     55 |
+------------------------+
1 row in set (0.00 sec)
```

A-1.4 Total Source Lines of Code (SLOC)

The output for the libodbc++ project for this query is

Total SLOC for libodbc++

```
+-------+
| SLOC  |
+-------+
| 16444 |
+-------+
1 row in set (0.11 sec)
```

for the log4cxx project the output is

Total SLOC for log4cxx

```
 +-------+
| SLOC  |
+-------+
| 45733 |
+-------+
1 row in set (0.20 sec)
```

and for the C++ Sockets Library the output is

Total SLOC for the C++ Sockets Library

```
+-------+
| SLOC  |
+-------+
| 26869 |
+-------+
1 row in set (0.13 sec)
```

A-1.5 The SLOC per File

The output for the libodbc++ project for this query is

The SLOC for each file in libodbc++

```
+----+------------------------+------+
| id | name                   | SLOC |
+----+------------------------+------+
|  1 | statement.cpp          |  786 |
|  2 | driverinfo.h           |  145 |
|  3 | resultsetmetadata.cpp  |  366 |
|  4 | threads.cpp            |  123 |
|  5 | errorhandler.cpp       |  302 |
|  6 | drivermanager.cpp      |  369 |
|  7 | datastream.cpp         |  293 |
|  8 | callablestatement.cpp  |   96 |
|  9 | dtconv.h               |  324 |
| 10 | datetime.cpp           |  316 |
| 11 | datahandler.h          |  279 |
| 12 | connection.cpp         |  504 |
| 13 | preparedstatement.cpp  |  486 |
| 14 | datahandler.cpp        | 1000 |
| 15 | databasemetadata.cpp   | 1766 |
| 16 | datastream.h           |  206 |
| 17 | driverinfo.cpp         |  135 |
```

```
| 18 | resultset.cpp          | 1552 |
| 19 | types.h                |  977 |
| 20 | resultsetmetadata.h    |  193 |
| 21 | config-win32.h         |  106 |
| 22 | statement.h            |  270 |
| 23 | drivermanager.h        |  164 |
| 24 | databasemetadata.h     | 1349 |
| 25 | threads.h              |   79 |
| 26 | callablestatement.h    |  166 |
| 27 | errorhandler.h         |  141 |
| 28 | config.h               |  173 |
| 28 | config.h               |  170 |
| 29 | connection.h           |  238 |
| 30 | preparedstatement.h    |  229 |
| 31 | resultset.h            |  629 |
| 32 | setup.h                |  220 |
| 33 | mainwindow.cpp         |  257 |
| 34 | resultwindow.h         |   83 |
| 35 | connectwindow.cpp      |  115 |
| 36 | connectwindow.h        |   60 |
| 37 | resultwindow.cpp       |  214 |
| 38 | main.cpp               |   35 |
| 39 | mainwindow.h           |   71 |
| 40 | isql++.cpp             | 1321 |
| 41 | isql++.h               |  136 |
+----+------------------------+------+
42 rows in set (0.11 sec)
```

for the log4cxx project the output is

The SLOC for each file in log4cxx

```
+-----+-----------------------------------------+------+
| id  | name                                    | SLOC |
+-----+-----------------------------------------+------+
|   1 | defaultconfigurator.h                   |   55 |
|   2 | odbcappender.h                          |  292 |
|   3 | rollingfileappender.h                   |  116 |
|   3 | rollingfileappender.h                   |  105 |
|   4 | ndc.h                                   |  353 |
|   5 | layout.h                                |   97 |
|   6 | hierarchy.h                             |  283 |
|   7 | htmllayout.h                            |  133 |
|   8 | basicconfigurator.h                     |   65 |
|   9 | file.h                                  |  188 |
|  10 | provisionnode.h                         |   35 |
```

15	11	xmllayout.h	138
16	12	domconfigurator.h	316
17	13	levelrangefilter.h	137
18	14	andfilter.h	109
19	15	mapfilter.h	44
20	16	expressionfilter.h	130
21	17	stringmatchfilter.h	92
22	18	levelmatchfilter.h	102
23	19	propertyfilter.h	81
24	20	denyallfilter.h	73
25	21	locationinfofilter.h	87
26	22	loggerfactory.h	46
27	23	rootlogger.h	60
28	24	hierarchyeventlistener.h	65
29	25	defaultrepositoryselector.h	50
30	26	optionhandler.h	72
31	27	triggeringeventevaluator.h	50
32	28	repositoryselector.h	54
33	29	appenderattachable.h	98
34	30	locationinfo.h	136
35	31	errorhandler.h	127
36	32	filter.h	133
37	33	loggerrepository.h	116
38	34	configurator.h	61
39	35	loggingevent.h	263
40	36	defaultloggerfactory.h	45
41	37	filewatchdog.h	86
42	38	iso8601dateformat.h	46
43	39	outputstream.h	59
44	40	stringhelper.h	61
45	41	dateformat.h	92
46	42	datelayout.h	91
47	43	inetaddress.h	106
48	44	optionconverter.h	164
49	45	bytearrayoutputstream.h	77
50	46	appenderattachableimpl.h	128
51	47	strictmath.h	49
52	48	cacheddateformat.h	220
53	49	transform.h	62
54	50	synchronized.h	44
55	51	class.h	67
56	52	locale.h	49
57	53	objectptr.h	183
58	54	xml.h	131
59	55	resourcebundle.h	89

60	56	reader.h	72
61	57	aprinitializer.h	57
62	58	socketoutputstream.h	74
63	59	strftimedateformat.h	68
64	60	condition.h	73
65	61	datagramsocket.h	128
66	62	simpledateformat.h	96
67	63	bufferedoutputstream.h	63
68	64	timezone.h	70
69	65	absolutetimedateformat.h	40
70	66	classregistration.h	41
71	67	properties.h	186
72	68	bytearrayinputstream.h	94
73	69	bufferedwriter.h	63
74	70	onlyonceerrorhandler.h	103
75	71	propertyresourcebundle.h	62
76	72	date.h	67
77	73	system.h	53
78	74	mutex.h	51
79	75	stringtokenizer.h	49
80	76	serversocket.h	64
81	77	outputstreamwriter.h	65
82	78	threadspecificdata.h	67
83	79	fileoutputstream.h	67
84	80	inputstream.h	72
85	81	socket.h	90
86	82	charsetdecoder.h	113
87	83	thread.h	184
88	84	systemerrwriter.h	59
89	85	objectimpl.h	49
90	86	integer.h	50
91	87	loglog.h	113
92	88	object.h	138
93	89	fileinputstream.h	96
94	90	transcoder.h	256
95	91	datagrampacket.h	135
96	92	cyclicbuffer.h	92
97	93	relativetimedateformat.h	47
98	94	datetimedateformat.h	42
99	95	syslogwriter.h	48
100	96	tchar.h	171
101	97	inputstreamreader.h	94
102	98	messagebuffer.h	828
103	99	loader.h	43
104	100	charsetencoder.h	135

101	bytebuffer.h	69
102	systemoutwriter.h	58
103	exception.h	287
104	objectoutputstream.h	95
105	pool.h	59
106	threadlocal.h	85
107	writer.h	58
108	log4cxx.h	51
109	rollingpolicy.h	82
111	rolloverdescription.h	101
112	filterbasedtriggeringpolicy.h	118
113	rollingpolicybase.h	128
114	action.h	87
115	timebasedrollingpolicy.h	227
116	triggeringpolicy.h	80
117	rollingfileappenderskeleton.h	152
118	manualtriggeringpolicy.h	72
119	filerenameaction.h	59
120	sizebasedtriggeringpolicy.h	82
121	gzcompressaction.h	73
122	zipcompressaction.h	74
123	fixedwindowrollingpolicy.h	141
124	log4cxx_private.h	56
125	methodlocationpatternconverter.h	61
126	patternparser.h	171
127	ndcpatternconverter.h	60
128	filedatepatternconverter.h	52
129	loggerpatternconverter.h	67
130	formattinginfo.h	117
131	fulllocationpatternconverter.h	63
132	loggingeventpatternconverter.h	84
133	threadpatternconverter.h	60
134	linelocationpatternconverter.h	62
135	levelpatternconverter.h	62
136	throwableinformationpatternconverter.h	74
137	lineseparatorpatternconverter.h	69
138	datepatternconverter.h	82
139	literalpatternconverter.h	67
140	namepatternconverter.h	74
141	nameabbreviator.h	81
142	classnamepatternconverter.h	65
143	filelocationpatternconverter.h	62
144	relativetimepatternconverter.h	59
145	patternconverter.h	124
146	propertiespatternconverter.h	71

147	integerpatternconverter.h	64
148	messagepatternconverter.h	60
149	asyncappender.h	292
150	portability.h	25
151	dailyrollingfileappender.h	203
152	logmanager.h	212
153	appender.h	148
154	fileappender.h	223
155	consoleappender.h	79
156	propertysetter.h	108
157	nteventlogappender.h	111
158	outputdebugstringappender.h	48
159	writerappender.h	217
160	socketappender.h	143
161	syslogappender.h	141
162	telnetappender.h	158
163	socketappenderskeleton.h	179
164	sockethubappender.h	197
165	smtpappender.h	283
166	xmlsocketappender.h	150
167	logstring.h	90
168	simplelayout.h	83
169	level.h	285
170	fallbackerrorhandler.h	116
171	appenderskeleton.h	213
172	mdc.h	240
173	ttcclayout.h	187
174	propertyconfigurator.h	395
175	stream.h	566
176	logger.h	1911
177	patternlayout.h	409
178	integer.cpp	34
179	locationinfo.cpp	195
180	properties.cpp	370
181	file.cpp	240
182	inputstreamreader.cpp	74
183	hierarchy.cpp	399
184	socket.cpp	130
185	socketappender.cpp	111
186	aprinitializer.cpp	69
187	propertiespatternconverter.cpp	80
188	ndc.cpp	338
189	threadpatternconverter.cpp	50
190	basicconfigurator.cpp	45
191	fulllocationpatternconverter.cpp	56

192	propertyconfigurator.cpp	451
193	nameabbreviator.cpp	325
194	objectoutputstream.cpp	194
195	outputstream.cpp	30
196	socketappenderskeleton.cpp	176
197	logstream.cpp	504
198	resourcebundle.cpp	122
199	relativetimepatternconverter.cpp	53
200	charsetencoder.cpp	518
201	inetaddress.cpp	144
202	throwableinformationpatternconverter.cpp	64
203	patternconverter.cpp	50
204	exception.cpp	427
205	rollingpolicybase.cpp	146
206	classregistration.cpp	28
207	stringmatchfilter.cpp	80
208	datagrampacket.cpp	58
209	datagramsocket.cpp	186
210	layout.cpp	40
211	threadlocal.cpp	60
212	levelmatchfilter.cpp	84
213	htmllayout.cpp	220
214	fallbackerrorhandler.cpp	106
215	writer.cpp	29
216	charsetdecoder.cpp	490
217	obsoleterollingfileappender.cpp	155
218	triggeringpolicy.cpp	35
219	fileoutputstream.cpp	98
220	patternparser.cpp	341
221	integerpatternconverter.cpp	52
222	systemerrwriter.cpp	73
223	transform.cpp	117
224	sizebasedtriggeringpolicy.cpp	56
225	onlyonceerrorhandler.cpp	89
226	outputstreamwriter.cpp	78
227	writerappender.cpp	264
228	datepatternconverter.cpp	135
229	class.cpp	192
230	telnetappender.cpp	252
231	rolloverdescription.cpp	62
232	nteventlogappender.cpp	316
233	dateformat.cpp	37
234	filterbasedtriggeringpolicy.cpp	95
235	objectptr.cpp	47
236	timezone.cpp	250

237	systemoutwriter.cpp	74
238	patternlayout.cpp	187
239	asyncappender.cpp	364
240	objectimpl.cpp	48
241	filedatepatternconverter.cpp	39
242	sockethubappender.cpp	238
243	filerenameaction.cpp	35
244	fileinputstream.cpp	94
245	zipcompressaction.cpp	86
246	system.cpp	96
247	logmanager.cpp	211
248	fixedwindowrollingpolicy.cpp	295
249	stringhelper.cpp	159
250	filewatchdog.cpp	93
251	cacheddateformat.cpp	309
252	appenderattachableimpl.cpp	160
253	reader.cpp	29
254	formattinginfo.cpp	68
255	odbcappender.cpp	333
256	methodlocationpatternconverter.cpp	51
257	bufferedwriter.cpp	60
258	bytearrayinputstream.cpp	55
259	strftimedateformat.cpp	55
260	syslogappender.cpp	356
261	rollingpolicy.cpp	24
262	rollingfileappender.cpp	384
263	loggingevent.cpp	326
264	gzcompressaction.cpp	95
265	namepatternconverter.cpp	56
266	outputdebugstringappender.cpp	47
267	loggingeventpatternconverter.cpp	49
268	simpledateformat.cpp	781
269	filelocationpatternconverter.cpp	50
270	threadspecificdata.cpp	131
271	xmlsocketappender.cpp	118
272	inputstream.cpp	30
273	optionconverter.cpp	381
274	appenderskeleton.cpp	177
275	date.cpp	53
276	locale.cpp	55
277	propertysetter.cpp	104
278	synchronized.cpp	49
279	propertyresourcebundle.cpp	57
280	bytebuffer.cpp	65
281	literalpatternconverter.cpp	62

282	simplelayout.cpp	37
283	loggerpatternconverter.cpp	55
284	levelrangefilter.cpp	89
285	configurator.cpp	32
286	ttcclayout.cpp	77
287	syslogwriter.cpp	68
288	manualtriggeringpolicy.cpp	42
289	messagebuffer.cpp	430
290	level.cpp	234
291	threadcxx.cpp	175
292	cyclicbuffer.cpp	148
293	loglog.cpp	121
294	fileappender.cpp	309
295	mutex.cpp	61
296	stringtokenizer.cpp	60
297	filter.cpp	49
298	datelayout.cpp	121
299	action.cpp	75
300	defaultloggerfactory.cpp	30
301	loader.cpp	71
302	domconfigurator.cpp	969
303	levelpatternconverter.cpp	87
304	relativetimedateformat.cpp	38
305	serversocket.cpp	140
306	timebasedrollingpolicy.cpp	193
307	defaultconfigurator.cpp	105
308	smtpappender.cpp	674
309	rootlogger.cpp	52
310	pool.cpp	86
311	socketoutputstream.cpp	57
312	messagepatternconverter.cpp	52
313	transcoder.cpp	538
314	dailyrollingfileappender.cpp	108
315	defaultrepositoryselector.cpp	40
316	condition.cpp	70
317	lineseparatorpatternconverter.cpp	57
318	classnamepatternconverter.cpp	56
319	logger.cpp	975
320	mdc.cpp	236
321	xmllayout.cpp	146
322	ndcpatternconverter.cpp	54
323	bytearrayoutputstream.cpp	53
324	consoleappender.cpp	146
325	linelocationpatternconverter.cpp	52

```
325 rows in set (0.00 sec)
```

and for the C++ Sockets Library the output is

The SLOC for each file in the C++ Sockets Library

```
+-----+-------------------------+------+
| id  | name                    | SLOC |
+-----+-------------------------+------+
|   1 | Event.h                 |   80 |
|   2 | Debug.cpp               |  103 |
|   3 | IHttpServer.h           |   68 |
|   4 | SocketStream.cpp        |   69 |
|   5 | HttpRequest.h           |  140 |
|   6 | EventTime.h             |   79 |
|   7 | HttpRequest.cpp         |  428 |
|   8 | HttpPostSocket.h        |   98 |
|   9 | Thread.cpp              |  178 |
|  10 | HttpBaseSocket.h        |   87 |
|  11 | MemFile.cpp             |  349 |
|  12 | Ajp13Socket.cpp         |  414 |
|  13 | HttpPostSocket.cpp      |  295 |
|  14 | XmlException.cpp        |   52 |
|  15 | SocketThread.cpp        |   51 |
|  16 | EventHandler.h          |   85 |
|  17 | Exception.h             |   71 |
|  18 | IEventOwner.h           |   79 |
|  19 | Lock.h                  |   61 |
|  20 | Sockets-config.cpp      |  121 |
|  21 | Ipv4Address.h           |  108 |
|  22 | ResolvServer.cpp        |   99 |
|  23 | StreamSocket.cpp        |  173 |
|  24 | SctpSocket.h            |  119 |
|  25 | HttpResponse.cpp        |  251 |
|  26 | EventHandler.cpp        |  209 |
|  27 | SocketStream.h          |   66 |
|  28 | SocketHandlerEp.cpp     |  216 |
|  29 | UdpSocket.cpp           |  905 |
|  30 | Mutex.cpp               |   83 |
|  31 | UdpSocket.h             |  224 |
|  32 | StdoutLog.cpp           |  108 |
|  33 | ResolvSocket.cpp        |  442 |
|  34 | StreamWriter.h          |   67 |
|  35 | HttpdCookies.cpp        |  286 |
|  36 | FileStream.h            |   61 |
|  37 | File.cpp                |  203 |
```

```
|  38 | Utility.cpp             | 1435 |
|  39 | XmlDocument.h           |   79 |
|  40 | sockets-config.h        |  110 |
|  41 | EventTime.cpp           |  107 |
|  42 | HTTPSocket.cpp          |  518 |
|  43 | SctpSocket.cpp          |  506 |
|  44 | SocketThread.h          |   40 |
|  45 | socket_include.cpp      |   93 |
|  46 | Parse.cpp               |  323 |
|  47 | IFileUpload.h           |   60 |
|  48 | AjpBaseSocket.h         |   95 |
|  49 | Debug.h                 |   78 |
|  50 | HttpdForm.h             |  141 |
|  51 | SocketHandlerThread.h   |   67 |
|  52 | SSLInitializer.cpp      |  181 |
|  53 | Ipv6Address.h           |  117 |
|  54 | SocketHandlerThread.cpp |   81 |
|  55 | SmtpdSocket.cpp         |  244 |
|  56 | XmlNode.h               |  150 |
|  57 | Thread.h                |  128 |
|  58 | Base64.cpp              |  314 |
|  59 | SocketHandler.h         |  295 |
|  60 | Lock.cpp                |   57 |
|  61 | XmlException.h          |   65 |
|  62 | HttpTransaction.cpp     |  262 |
|  63 | FileStream.cpp          |   60 |
|  64 | Base64.h                |   82 |
|  65 | TcpSocket.cpp           | 1850 |
|  66 | StreamWriter.cpp        |   92 |
|  67 | SocketHandler.cpp       | 1418 |
|  68 | Json.h                  |   96 |
|  69 | Mutex.h                 |   72 |
|  70 | Exception.cpp           |   66 |
|  71 | IStream.h               |   60 |
|  72 | HttpBaseSocket.cpp      |  215 |
|  73 | HTTPSocket.h            |  168 |
|  74 | IFile.h                 |   74 |
|  75 | ListenSocket.h          |  512 |
|  76 | ResolvSocket.h          |  109 |
|  77 | HttpResponse.h          |   98 |
|  78 | Semaphore.h             |   97 |
|  79 | IEventOwner.cpp         |   81 |
|  80 | IBase.h                 |   44 |
|  81 | TcpSocket.h             |  365 |
|  82 | Semaphore.cpp           |  123 |
```

```
|  83 | XmlDocument.cpp         |  102 |
|  84 | HttpDebugSocket.h       |   73 |
|  85 | AjpBaseSocket.cpp       |  268 |
|  86 | SSLInitializer.h        |   96 |
|  87 | HttpDebugSocket.cpp     |  144 |
|  88 | socket_include.h        |  298 |
|  89 | SocketAddress.h         |  105 |
|  90 | StdLog.h                |   76 |
|  91 | HttpdSocket.h           |  108 |
|  92 | HttpPutSocket.h         |   85 |
|  93 | ISocketHandler.h        |  261 |
|  94 | HttpdCookies.h          |   91 |
|  95 | ResolvServer.h          |   75 |
|  96 | Ipv6Address.cpp         |  285 |
|  97 | XmlNode.cpp             |  357 |
|  98 | MemFile.h               |  115 |
|  99 | SmtpdSocket.h           |  153 |
| 100 | HttpGetSocket.h         |   68 |
| 101 | HttpClientSocket.h      |  134 |
| 102 | Ajp13Socket.h           |   80 |
| 103 | Event.cpp               |   93 |
| 104 | HttpPutSocket.cpp       |  125 |
| 105 | HttpdSocket.cpp         |  326 |
| 106 | HttpTransaction.h       |  105 |
| 107 | Ipv4Address.cpp         |  223 |
| 108 | IEventHandler.h         |   77 |
| 109 | Socket.h                |  720 |
| 110 | HttpClientSocket.cpp    |  312 |
| 111 | IMutex.h                |   57 |
| 112 | Parse.h                 |  103 |
| 113 | Json.cpp                |  558 |
| 114 | Socket.cpp              | 1875 |
| 115 | StreamSocket.h          |  139 |
| 116 | StdoutLog.h             |   63 |
| 117 | HttpdForm.cpp           |  716 |
| 118 | File.h                  |   92 |
| 119 | ajp13.h                 |  101 |
| 120 | Utility.h               |  245 |
| 121 | SocketHandlerEp.h       |   95 |
| 122 | HttpGetSocket.cpp       |  114 |
+-----+-------------------------+------+
122 rows in set (0.14 sec)
```

A-1.6 Total Header SLOC

The output for the libodbc++ project for this query is

Total Header SLOC for libodbc++

```
+--------------------------+
| The SLOC in Header Files |
+--------------------------+
|                     6408 |
+--------------------------+
1 row in set (0.04 sec)
```

for the log4cxx project the output is

Total Header SLOC for log4cxx

```
+--------------------------+
| The SLOC in Header Files |
+--------------------------+
|                    22356 |
+--------------------------+
1 row in set (0.07 sec)
```

and for the C++ Sockets Library the output is

Total Header SLOC for the C++ Sockets Library

```
+--------------------------+
| The SLOC in Header Files |
+--------------------------+
|                     8310 |
+--------------------------+
1 row in set (0.04 sec)
```

A-1.7 Total Source SLOC

The output for the libodbc++ project for this query is

Total Source SLOC for libodbc++

```
+--------------------------+
| The SLOC in Source Files |
+--------------------------+
|                    10036 |
+--------------------------+
1 row in set (0.07 sec)
```

for the log4cxx project the output is

Total Source SLOC for log4cxx

```
+--------------------------+
| The SLOC in Source Files |
+--------------------------+
|                    23377 |
+--------------------------+
1 row in set (0.04 sec)
```

and for the C++ Sockets Library the output is

Total Source SLOC for the C++ Sockets Library

```
+--------------------------+
| The SLOC in Header Files |
+--------------------------+
|                    18559 |
+--------------------------+
1 row in set (0.13 sec)
```

A-1.8 The Bytes per File

The output for the libodbc++ project for this query is

The Bytes for each File in libodbc++

```
+-----------------------+------------+
| name                  | Byte Count |
+-----------------------+------------+
| statement.cpp         |      22497 |
| driverinfo.h          |       3819 |
| resultsetmetadata.cpp |       9413 |
| threads.cpp           |       2539 |
| errorhandler.cpp      |       7824 |
| drivermanager.cpp     |       9233 |
```

```
| datastream.cpp        |       7084 |
| callablestatement.cpp |       2891 |
| dtconv.h              |       8183 |
| datetime.cpp          |       7173 |
| datahandler.h         |       6625 |
| connection.cpp        |      13910 |
| preparedstatement.cpp |      12716 |
| datahandler.cpp       |      24347 |
| databasemetadata.cpp  |      43360 |
| datastream.h          |       4999 |
| driverinfo.cpp        |       4941 |
| resultset.cpp         |      39669 |
| types.h               |      23812 |
| resultsetmetadata.h   |       5724 |
| config-win32.h        |       2154 |
| statement.h           |       8965 |
| drivermanager.h       |       4777 |
| databasemetadata.h    |      49091 |
| threads.h             |       1760 |
| callablestatement.h   |       5052 |
| errorhandler.h        |       3964 |
| config.h              |       5111 |
| config.h              |       4607 |
| connection.h          |       7484 |
| preparedstatement.h   |       6936 |
| resultset.h           |      18732 |
| setup.h               |       6032 |
| mainwindow.cpp        |       5895 |
| resultwindow.h        |       1797 |
| connectwindow.cpp     |       2911 |
| connectwindow.h       |       1479 |
| resultwindow.cpp      |       4405 |
| main.cpp              |       1115 |
| mainwindow.h          |       1698 |
| isql++.cpp            |      30837 |
| isql++.h              |       2569 |
+-----------------------+------------+
42 rows in set (0.00 sec)
```

for the log4cxx project the output is

The Bytes for each File in log4cxx

```
+-----------------------------------------+------------+
| name                                    | Byte Count |
+-----------------------------------------+------------+
```

defaultconfigurator.h	1747
odbcappender.h	12177
rollingfileappender.h	4474
rollingfileappender.h	4069
ndc.h	15652
layout.h	3444
hierarchy.h	10162
htmllayout.h	4749
basicconfigurator.h	2202
file.h	7377
provisionnode.h	1168
xmllayout.h	5638
domconfigurator.h	13966
levelrangefilter.h	6075
andfilter.h	3710
mapfilter.h	1356
expressionfilter.h	4363
stringmatchfilter.h	4076
levelmatchfilter.h	4153
propertyfilter.h	2506
denyallfilter.h	2768
locationinfofilter.h	3087
loggerfactory.h	1663
rootlogger.h	2213
hierarchyeventlistener.h	2243
defaultrepositoryselector.h	2028
optionhandler.h	2966
triggeringeventevaluator.h	2029
repositoryselector.h	2160
appenderattachable.h	2945
locationinfo.h	3700
errorhandler.h	5154
filter.h	5201
loggerrepository.h	4456
configurator.h	1905
loggingevent.h	11381
defaultloggerfactory.h	1698
filewatchdog.h	3105
iso8601dateformat.h	1790
outputstream.h	1972
stringhelper.h	2309
dateformat.h	3416
datelayout.h	3762
inetaddress.h	3775
optionconverter.h	7221

bytearrayoutputstream.h	2440
appenderattachableimpl.h	4101
strictmath.h	1837
cacheddateformat.h	8194
transform.h	2829
synchronized.h	1542
class.h	2199
locale.h	1927
objectptr.h	5610
xml.h	4916
resourcebundle.h	3648
reader.h	2275
aprinitializer.h	1754
socketoutputstream.h	2589
strftimedateformat.h	2236
condition.h	2874
datagramsocket.h	5290
simpledateformat.h	3132
bufferedoutputstream.h	2198
timezone.h	2064
absolutetimedateformat.h	1545
classregistration.h	1539
properties.h	9067
bytearrayinputstream.h	3104
bufferedwriter.h	2175
onlyonceerrorhandler.h	3895
propertyresourcebundle.h	2534
date.h	1790
system.h	1838
mutex.h	1629
stringtokenizer.h	1871
serversocket.h	2278
outputstreamwriter.h	2358
threadspecificdata.h	2547
fileoutputstream.h	2404
inputstream.h	2485
socket.h	3477
charsetdecoder.h	4033
thread.h	7856
systemerrwriter.h	2069
objectimpl.h	1542
integer.h	1387
loglog.h	4483
object.h	5200
fileinputstream.h	3475

transcoder.h	8350
datagrampacket.h	6167
cyclicbuffer.h	3873
relativetimedateformat.h	1814
datetimedateformat.h	1725
syslogwriter.h	1718
tchar.h	7344
inputstreamreader.h	3406
messagebuffer.h	26003
loader.h	1534
charsetencoder.h	4585
bytebuffer.h	2243
systemoutwriter.h	2094
exception.h	13439
objectoutputstream.h	3647
pool.h	2012
threadlocal.h	2887
writer.h	1871
log4cxx.h	1560
rollingpolicy.h	3045
rolloverdescription.h	3476
filterbasedtriggeringpolicy.h	3195
rollingpolicybase.h	3927
action.h	2285
timebasedrollingpolicy.h	9684
triggeringpolicy.h	2794
rollingfileappenderskeleton.h	5120
manualtriggeringpolicy.h	2590
filerenameaction.h	1814
sizebasedtriggeringpolicy.h	2784
gzcompressaction.h	2099
zipcompressaction.h	2119
fixedwindowrollingpolicy.h	5051
log4cxx_private.h	1770
methodlocationpatternconverter.h	1906
patternparser.h	5421
ndcpatternconverter.h	1792
filedatepatternconverter.h	1566
loggerpatternconverter.h	1934
formattinginfo.h	2665
fulllocationpatternconverter.h	1863
loggingeventpatternconverter.h	2738
threadpatternconverter.h	1830
linelocationpatternconverter.h	1862
levelpatternconverter.h	1886

139	throwableinformationpatternconverter.h	2353
140	lineseparatorpatternconverter.h	1965
141	datepatternconverter.h	2480
142	literalpatternconverter.h	1898
143	namepatternconverter.h	2215
144	nameabbreviator.h	2467
145	classnamepatternconverter.h	1991
146	filelocationpatternconverter.h	1884
147	relativetimepatternconverter.h	1886
148	patternconverter.h	3537
149	propertiespatternconverter.h	2405
150	integerpatternconverter.h	1807
151	messagepatternconverter.h	1813
152	asyncappender.h	10363
153	portability.h	926
154	dailyrollingfileappender.h	6499
155	logmanager.h	7088
156	appender.h	4571
157	fileappender.h	8843
158	consoleappender.h	2887
159	propertysetter.h	4657
160	nteventlogappender.h	4621
161	outputdebugstringappender.h	1804
162	writerappender.h	8255
163	socketappender.h	5947
164	syslogappender.h	6429
165	telnetappender.h	6205
166	socketappenderskeleton.h	6824
167	sockethubappender.h	9015
168	smtpappender.h	12128
169	xmlsocketappender.h	6363
170	logstring.h	2306
171	simplelayout.h	2829
172	level.h	10963
173	fallbackerrorhandler.h	4642
174	appenderskeleton.h	7552
175	mdc.h	10342
176	ttcclayout.h	7183
177	propertyconfigurator.h	13543
178	stream.h	20771
179	logger.h	74966
180	patternlayout.h	14232
181	integer.cpp	1069
182	locationinfo.cpp	5809
183	properties.cpp	14490

file.cpp	5977
inputstreamreader.cpp	2362
hierarchy.cpp	10880
socket.cpp	3970
socketappender.cpp	3302
aprinitializer.cpp	1959
propertiespatternconverter.cpp	2760
ndc.cpp	7610
threadpatternconverter.cpp	1748
basicconfigurator.cpp	1655
fulllocationpatternconverter.cpp	2056
propertyconfigurator.cpp	16068
nameabbreviator.cpp	8867
objectoutputstream.cpp	6480
outputstream.cpp	1046
socketappenderskeleton.cpp	6239
logstream.cpp	13956
resourcebundle.cpp	3489
relativetimepatternconverter.cpp	1891
charsetencoder.cpp	18715
inetaddress.cpp	4140
throwableinformationpatternconverter.cpp	2275
patternconverter.cpp	1604
exception.cpp	11625
rollingpolicybase.cpp	4609
classregistration.cpp	1085
stringmatchfilter.cpp	2232
datagrampacket.cpp	2138
datagramsocket.cpp	5266
layout.cpp	1321
threadlocal.cpp	1823
levelmatchfilter.cpp	2338
htmllayout.cpp	8432
fallbackerrorhandler.cpp	3698
writer.cpp	985
charsetdecoder.cpp	15992
obsoleterollingfileappender.cpp	5102
triggeringpolicy.cpp	1177
fileoutputstream.cpp	2851
patternparser.cpp	10130
integerpatternconverter.cpp	1741
systemerrwriter.cpp	1932
transform.cpp	3138
sizebasedtriggeringpolicy.cpp	1954
onlyonceerrorhandler.cpp	2345

outputstreamwriter.cpp	2425
writerappender.cpp	6840
datepatternconverter.cpp	4422
class.cpp	6249
telnetappender.cpp	8295
rolloverdescription.cpp	1909
nteventlogappender.cpp	9785
dateformat.cpp	1237
filterbasedtriggeringpolicy.cpp	2658
objectptr.cpp	1637
timezone.cpp	7167
systemoutwriter.cpp	1937
patternlayout.cpp	6237
asyncappender.cpp	11320
objectimpl.cpp	1319
filedatepatternconverter.cpp	1490
sockethubappender.cpp	7798
filerenameaction.cpp	1333
fileinputstream.cpp	2678
zipcompressaction.cpp	2934
system.cpp	3083
logmanager.cpp	5844
fixedwindowrollingpolicy.cpp	8202
stringhelper.cpp	4623
filewatchdog.cpp	2600
cacheddateformat.cpp	9785
appenderattachableimpl.cpp	4072
reader.cpp	985
formattinginfo.cpp	2279
odbcappender.cpp	8547
methodlocationpatternconverter.cpp	1809
bufferedwriter.cpp	1690
bytearrayinputstream.cpp	1739
strftimedateformat.cpp	1783
syslogappender.cpp	11916
rollingpolicy.cpp	927
rollingfileappender.cpp	10423
loggingevent.cpp	10395
gzcompressaction.cpp	3364
namepatternconverter.cpp	1965
outputdebugstringappender.cpp	1501
loggingeventpatternconverter.cpp	1644
simpledateformat.cpp	19000
filelocationpatternconverter.cpp	1810
threadspecificdata.cpp	3643

xmlsocketappender.cpp	3609
inputstream.cpp	1040
optionconverter.cpp	13129
appenderskeleton.cpp	4552
date.cpp	1419
locale.cpp	1462
propertysetter.cpp	3596
synchronized.cpp	1586
propertyresourcebundle.cpp	1752
bytebuffer.cpp	1682
literalpatternconverter.cpp	2073
simplelayout.cpp	1338
loggerpatternconverter.cpp	1939
levelrangefilter.cpp	2877
configurator.cpp	1054
ttcclayout.cpp	2589
syslogwriter.cpp	2171
manualtriggeringpolicy.cpp	1543
messagebuffer.cpp	14451
level.cpp	6093
threadcxx.cpp	4618
cyclicbuffer.cpp	4029
loglog.cpp	3273
fileappender.cpp	9058
mutex.cpp	1786
stringtokenizer.cpp	1932
filter.cpp	1347
datelayout.cpp	4177
action.cpp	1831
defaultloggerfactory.cpp	1140
loader.cpp	2434
domconfigurator.cpp	39492
levelpatternconverter.cpp	2568
relativetimedateformat.cpp	1418
serversocket.cpp	3891
timebasedrollingpolicy.cpp	5555
defaultconfigurator.cpp	4008
smtpappender.cpp	20605
rootlogger.cpp	1470
pool.cpp	2212
socketoutputstream.cpp	1725
messagepatternconverter.cpp	1744
transcoder.cpp	15115
dailyrollingfileappender.cpp	3089
defaultrepositoryselector.cpp	1342

```
| condition.cpp                              |        1968 |
| lineseparatorpatternconverter.cpp          |        1944 |
| classnamepatternconverter.cpp              |        2016 |
| logger.cpp                                 |       24986 |
| mdc.cpp                                    |        6219 |
| xmllayout.cpp                              |        6325 |
| ndcpatternconverter.cpp                    |        1741 |
| bytearrayoutputstream.cpp                  |        1622 |
| consoleappender.cpp                        |        4133 |
| linelocationpatternconverter.cpp           |        1866 |
+--------------------------------------------+-------------+
325 rows in set (0.00 sec)
```

and for the C++ Sockets Library the output is

The Bytes for each File in the C++ Sockets Library

```
+--------------------------+-------------+
| name                     | Byte Count  |
+--------------------------+-------------+
| Event.h                  |        2081 |
| Debug.cpp                |        1816 |
| IHttpServer.h            |        1917 |
| SocketStream.cpp         |        1920 |
| HttpRequest.h            |        3694 |
| EventTime.h              |        2166 |
| HttpRequest.cpp          |        9981 |
| HttpPostSocket.h         |        3111 |
| Thread.cpp               |        3364 |
| HttpBaseSocket.h         |        2293 |
| MemFile.cpp              |        6486 |
| Ajp13Socket.cpp          |        9945 |
| HttpPostSocket.cpp       |        7599 |
| XmlException.cpp         |        1513 |
| SocketThread.cpp         |         873 |
| EventHandler.h           |        2421 |
| Exception.h              |        1907 |
| IEventOwner.h            |        2255 |
| Lock.h                   |        1606 |
| Sockets-config.cpp       |        2807 |
| Ipv4Address.h            |        3230 |
| ResolvServer.cpp         |        2080 |
| StreamSocket.cpp         |        2984 |
| SctpSocket.h             |        3271 |
| HttpResponse.cpp         |        6427 |
| EventHandler.cpp         |        4420 |
```

SocketStream.h	1825
SocketHandlerEp.cpp	4566
UdpSocket.cpp	19767
Mutex.cpp	1891
UdpSocket.h	7419
StdoutLog.cpp	2619
ResolvSocket.cpp	9859
StreamWriter.h	1868
HttpdCookies.cpp	6842
FileStream.h	1674
File.cpp	3358
Utility.cpp	34827
XmlDocument.h	2043
sockets-config.h	2588
EventTime.cpp	2257
HTTPSocket.cpp	11150
SctpSocket.cpp	11375
SocketThread.h	684
socket_include.cpp	3669
Parse.cpp	6235
IFileUpload.h	1711
AjpBaseSocket.h	2556
Debug.h	1805
HttpdForm.h	4010
SocketHandlerThread.h	1737
SSLInitializer.cpp	3829
Ipv6Address.h	3354
SocketHandlerThread.cpp	1905
SmtpdSocket.cpp	4571
XmlNode.h	4462
Thread.h	3178
Base64.cpp	7841
SocketHandler.h	9329
Lock.cpp	1485
XmlException.h	1804
HttpTransaction.cpp	5948
FileStream.cpp	1679
Base64.h	2318
TcpSocket.cpp	38044
StreamWriter.cpp	2190
SocketHandler.cpp	29905
Json.h	2120
Mutex.h	1840
Exception.cpp	1739
IStream.h	1788

HttpBaseSocket.cpp	5153
HTTPSocket.h	5868
IFile.h	2103
ListenSocket.h	13161
ResolvSocket.h	3051
HttpResponse.h	2737
Semaphore.h	2312
IEventOwner.cpp	1915
IBase.h	1056
TcpSocket.h	11514
Semaphore.cpp	2473
XmlDocument.cpp	2465
HttpDebugSocket.h	2162
AjpBaseSocket.cpp	7789
SSLInitializer.h	2276
HttpDebugSocket.cpp	3457
socket_include.h	6491
SocketAddress.h	3045
StdLog.h	1937
HttpdSocket.h	3206
HttpPutSocket.h	2445
ISocketHandler.h	8880
HttpdCookies.h	2662
ResolvServer.h	2002
Ipv6Address.cpp	5841
XmlNode.cpp	6722
MemFile.h	2963
SmtpdSocket.h	4063
HttpGetSocket.h	2038
HttpClientSocket.h	4429
Ajp13Socket.h	2260
Event.cpp	1936
HttpPutSocket.cpp	2880
HttpdSocket.cpp	6740
HttpTransaction.h	2957
Ipv4Address.cpp	4682
IEventHandler.h	2311
Socket.h	23249
HttpClientSocket.cpp	5855
IMutex.h	1612
Parse.h	2606
Json.cpp	11771
Socket.cpp	36527
StreamSocket.h	4171
StdoutLog.h	1821

```
| HttpdForm.cpp           |       14870 |
| File.h                  |        2330 |
| ajp13.h                 |        3254 |
| Utility.h               |        7510 |
| SocketHandlerEp.h       |        2622 |
| HttpGetSocket.cpp       |        3099 |
+-------------------------+-------------+
122 rows in set (0.00 sec)
```

A-1.9 The Number of Functions

The output for the libodbc++ project for this query is

The Number of Functions in libodbc++

```
+-----------------+
| Function Count |
+-----------------+
|            1298 |
+-----------------+
1 row in set (0.00 sec)
```

for the log4cxx project the output is

The Number of Functions in log4cxx

```
+-----------------+
| Function Count |
+-----------------+
|            1680 |
+-----------------+
1 row in set (3.19 sec)
```

and for the C++ Sockets Library the output is

The Number of Functions in the C++ Sockets Library

```
+-----------------+
| Function Count |
+-----------------+
|            3331 |
+-----------------+
1 row in set (0.90 sec)
```

A-1.10 The Number of Functions per Generated XML File

The output for the libodbc++ project for this query is

The Functions per Generated XML File in libodbc++

```
+--------+-----------------------------+
| xml_id | Function Count per XML File |
+--------+-----------------------------+
|      1 |                        2194 |
|      2 |                        1834 |
|      3 |                        1839 |
|      4 |                        1052 |
|      5 |                        1837 |
|      6 |                        1840 |
|      7 |                        2192 |
|      8 |                        1272 |
|      9 |                        1276 |
|     10 |                        2554 |
|     11 |                        1837 |
|     12 |                        2193 |
|     13 |                        2196 |
|     14 |                         917 |
|     16 |                         134 |
|     17 |                         134 |
|     18 |                        2082 |
|     19 |                        2436 |
|     20 |                        2082 |
|     21 |                        1043 |
|     22 |                        2544 |
+--------+-----------------------------+
21 rows in set (0.00 sec)
```

for the log4cxx project the output is

The Functions per Generated XML File in log4cxx

```
+--------+-----------------------------+
| xml_id | Function Count per XML File |
+--------+-----------------------------+
|      1 |                        1545 |
|      2 |                        1790 |
|      3 |                        2433 |
|      4 |                        1331 |
|      5 |                        1544 |
|      6 |                        2200 |
|      7 |                        1799 |
```

```
|       8 |                        1544 |
|       9 |                        2003 |
|      10 |                         770 |
|      11 |                        1546 |
|      12 |                        2012 |
|      13 |                        1544 |
|      14 |                         770 |
|      15 |                         770 |
|      16 |                         770 |
|      17 |                        1544 |
|      18 |                        1544 |
|      19 |                         770 |
|      20 |                         770 |
|      21 |                         770 |
|      22 |                         770 |
|      23 |                        1540 |
|      24 |                         673 |
|      25 |                        1540 |
|      26 |                         667 |
|      27 |                         770 |
|      28 |                         667 |
|      29 |                         673 |
|      30 |                        1580 |
|      31 |                         770 |
|      32 |                        1540 |
|      33 |                         770 |
|      34 |                        1547 |
|      35 |                        2284 |
|      36 |                        1544 |
|      37 |                        2065 |
|      38 |                         673 |
|      39 |                        1338 |
|      40 |                        1703 |
|      41 |                        1344 |
|      42 |                        1540 |
|      43 |                        1880 |
|      44 |                        1565 |
|      45 |                        1542 |
|      46 |                        1581 |
|      47 |                         134 |
|      48 |                        1569 |
|      49 |                        1325 |
|      50 |                        1017 |
|      51 |                        1539 |
|      52 |                        1320 |
```

```
|      53 |                        1028 |
|      54 |                         667 |
|      55 |                        1445 |
|      56 |                        1338 |
|      57 |                        1201 |
|      58 |                         673 |
|      59 |                        1566 |
|      60 |                        1036 |
|      61 |                        1885 |
|      62 |                        1746 |
|      63 |                         667 |
|      64 |                        1818 |
|      65 |                         673 |
|      66 |                         801 |
|      67 |                        1540 |
|      68 |                        1933 |
|      69 |                        1338 |
|      70 |                        1544 |
|      71 |                        1547 |
|      72 |                        1570 |
|      73 |                        1782 |
|      74 |                        1314 |
|      75 |                        1323 |
|      76 |                        1891 |
|      77 |                        1465 |
|      78 |                        1856 |
|      79 |                        2061 |
|      80 |                        1338 |
|      81 |                        1885 |
|      82 |                        2071 |
|      83 |                         660 |
|      84 |                        1456 |
|      85 |                        1859 |
|      86 |                        1338 |
|      87 |                        1942 |
|      88 |                         667 |
|      89 |                        2061 |
|      90 |                        1819 |
|      91 |                        1350 |
|      92 |                        1543 |
|      93 |                        1669 |
|      94 |                         673 |
|      95 |                        1348 |
|      97 |                         667 |
|      98 |                        1514 |
```

99	1577
100	2071
101	1254
102	1456
103	1400
104	1676
105	1863
106	1837
107	1338
108	134
109	1544
110	1338
111	1544
112	1547
113	1340
114	1779
115	1544
116	770
117	1544
118	1544
119	1544
120	2084
121	2084
122	1547
124	1545
125	1443
126	1545
127	1443
128	1545
129	1339
130	1545
131	1544
132	1545
133	1545
134	1545
135	1546
136	1545
137	1544
138	1545
139	1544
140	1351
141	1545
142	1545
143	1545
144	1350

```
|     145 |                           1545 |
|     146 |                           1351 |
|     147 |                           1545 |
|     148 |                           2083 |
|     149 |                            134 |
|     150 |                           1544 |
|     151 |                           1965 |
|     152 |                            673 |
|     153 |                           1664 |
|     154 |                           1546 |
|     155 |                           1430 |
|     156 |                            904 |
|     157 |                            904 |
|     158 |                           1544 |
|     159 |                           2069 |
|     160 |                           1549 |
|     161 |                           2092 |
|     162 |                           1540 |
|     163 |                           2069 |
|     164 |                           1788 |
|     165 |                           1776 |
|     166 |                            660 |
|     167 |                           1544 |
|     168 |                           1866 |
|     169 |                           1544 |
|     170 |                           1775 |
|     171 |                           1336 |
|     172 |                           1546 |
|     173 |                           2012 |
|     174 |                            770 |
|     175 |                           2061 |
|     176 |                           1544 |
|     177 |                            770 |
|     178 |                            774 |
|     179 |                           1192 |
+---------+-------------------------------+
177 rows in set (3 min 20.29 sec)
```

and for the C++ Sockets Library the output is

The Functions per Generated XML File in the C++ Sockets Library

```
+---------+-------------------------------+
| xml_id | Function Count per XML File |
+---------+-------------------------------+
|       1 |                           302 |
```

2	2126
3	134
4	1728
5	2144
6	302
7	6390
8	624
9	6324
10	1570
11	6324
12	268
13	6324
14	6338
15	1890
16	268
17	268
18	2026
19	1068
20	6310
21	6310
22	1890
23	6324
24	6310
25	594
26	1320
27	6324
28	1890
29	1320
30	1320
31	1556
32	1890
33	268
34	6324
35	722
36	1320
37	660
38	6324
39	2124
40	6324
41	6152
42	268
43	6324
44	1890
45	1556
46	6332

```
|      47 |                            1890 |
|      48 |                            6324 |
|      49 |                            2326 |
|      50 |                             660 |
|      51 |                             660 |
|      52 |                            3163 |
|      53 |                             624 |
|      54 |                             134 |
|      55 |                            6324 |
|      56 |                             864 |
|      57 |                             660 |
|      58 |                            6324 |
|      59 |                            6390 |
|      60 |                            3162 |
|      61 |                            6326 |
|      62 |                            6324 |
|      63 |                             134 |
|      64 |                            6568 |
|      65 |                             134 |
|      66 |                             134 |
+---------+--------------------------------+
66 rows in set (0.00 sec)
```

A-1.11 Total Class Count

The output for the libodbc++ project for this query is

Total Class Count for libodbc++

```
+--------------+
| Class Count |
+--------------+
|          361 |
+--------------+
1 row in set (0.05 sec)
```

for the log4cxx project the output is

Total Class Count for log4cxx

```
+-------------+
| Class Count |
+-------------+
|         752 |
+-------------+
1 row in set (0.49 sec)
```

and for the C++ Sockets Library the output is

Total Class Count for the C++ Sockets Library

```
+-------------+
| Class Count |
+-------------+
|         280 |
+-------------+
1 row in set (0.04 sec)
```

A-1.12 Total Variable Count

The output for the libodbc++ project for this query is

The Total Variable Count for libodbc++

```
+----------------+
| Variable Count |
+----------------+
|            392 |
+----------------+
1 row in set (0.06 sec)
```

for the log4cxx project the output is

The Total Variable Count for log4cxx

```
+----------------+
| Variable Count |
+----------------+
|           1050 |
+----------------+
1 row in set (0.61 sec)
```

and for the C++ Sockets Library the output is

The Project Variable Count for the C++ Sockets Library

```
+----------------+
| Variable Count |
+----------------+
|            285 |
+----------------+
1 row in set (0.04 sec)
```

A-1.13 Total Private Variable Count

The output for the libodbc++ project for this query is

Total Private Variable Count for libodbc++

```
+------------------------+
| Private Variable Count |
+------------------------+
|                     81 |
+------------------------+
1 row in set (0.02 sec)
```

for the log4cxx project the output is

Total Private Variable Count for log4cxx

```
+------------------------+
| Private Variable Count |
+------------------------+
|                     23 |
+------------------------+
1 row in set (0.04 sec)
```

and for the C++ Sockets Library the output is

The Private Variable Count for the C++ Sockets Library

```
+------------------------+
| Private Variable Count |
+------------------------+
|                     40 |
+------------------------+
1 row in set (0.01 sec)
```

A-1.14 Total Protected Variable Count

The output for the libodbc++ project for this query is

Total Protected Variable Count for libodbc++

```
+--------------------------+
| Protected Variable Count |
+--------------------------+
|                        0 |
+--------------------------+
1 row in set (0.01 sec)
```

for the log4cxx project the output is

Total Protected Variable Count for log4cxx

```
+--------------------------+
| Protected Variable Count |
+--------------------------+
|                        0 |
+--------------------------+
1 row in set (0.03 sec)
```

and for the C++ Sockets Library the output is

Total Protected Variable Count for the C++ Sockets Library

```
+--------------------------+
| Protected Variable Count |
+--------------------------+
|                        1 |
+--------------------------+
1 row in set (0.01 sec)
```

A-1.15 Total Public Variable Count

The output for the libodbc++ project for this query is

Total Public Variable Count for libodbc++

```
+------------------------+
| Public Variable Count |
+------------------------+
|                    311 |
+------------------------+
1 row in set (0.06 sec)
```

for the log4cxx project the output is

Total Public Variable Count for log4cxx

```
+------------------------+
| Public Variable Count |
+------------------------+
|                   1029 |
+------------------------+
1 row in set (0.59 sec)
```

and for the C++ Sockets Library the output is

Total Public Variable Count for the C++ Sockets Library

```
+------------------------+
| Public Variable Count |
+------------------------+
|                    244 |
+------------------------+
1 row in set (0.05 sec)
```

A-1.16 List of Included Library Files

The output for the libodbc++ project for this query is

The List of Included Library Files for libodbc++

```
+-------------------------------------------------------------------------+
| name                                                                    |
+-------------------------------------------------------------------------+
| /usr/include/time.h                                                     |
| /usr/include/string.h                                                   |
| /opt/projects/gccxml/install/share/gccxml-0.9/GCC/4.4/gccxml_builtins.h |
| /usr/include/sqlext.h                                                   |
| /usr/include/sqltypes.h                                                 |
| /usr/include/unistd.h                                                   |
```

```
| /usr/include/sql.h                                                  |
| /usr/include/pthread.h                                              |
| /usr/include/bits/time.h                                            |
| /usr/include/sqlucode.h                                             |
| /usr/include/wchar.h                                                |
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/gthr-default.h           |
| /usr/include/stdio.h                                                |
| /usr/include/stdlib.h                                               |
| /usr/include/stdint.h                                               |
| /usr/include/sched.h                                                |
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/atomic_word.h            |
| /usr/include/bits/pthreadtypes.h                                    |
| /usr/include/c++/4.4/exception                                      |
| /usr/include/wctype.h                                               |
| /usr/include/c++/4.4/iosfwd                                         |
| /usr/include/bits/types.h                                           |
| /usr/include/ctype.h                                                |
| /usr/include/sys/types.h                                            |
| /usr/include/c++/4.4/new                                            |
| /usr/include/bits/sigset.h                                          |
| /usr/include/signal.h                                               |
| /usr/include/libio.h                                                |
| /usr/include/c++/4.4/bits/locale_facets.h                           |
| /usr/include/inttypes.h                                             |
| /usr/include/c++/4.4/bits/locale_classes.h                          |
| /usr/include/_G_config.h                                            |
| /usr/include/bits/waitstatus.h                                      |
| /usr/include/bits/confname.h                                        |
| /usr/include/assert.h                                               |
| /usr/include/xlocale.h                                              |
| /usr/include/sys/sysmacros.h                                        |
| /usr/include/bits/sched.h                                           |
| /usr/include/getopt.h                                               |
| /usr/include/sys/select.h                                           |
| /usr/include/bits/sys_errlist.h                                     |
| /usr/include/locale.h                                               |
| /usr/include/bits/setjmp.h                                          |
| /usr/include/alloca.h                                               |
| /usr/include/c++/4.4/bits/ios_base.h                                |
| /usr/lib/gcc/x86_64-linux-gnu/4.4.5/include/stddef.h                |
| /usr/include/c++/4.4/cxxabi-forced.h                                |
| /usr/lib/gcc/x86_64-linux-gnu/4.4.5/include/stdarg.h                |
| /usr/include/bits/locale.h                                          |
| /usr/include/c++/4.4/bits/stl_bvector.h                             |
| /usr/include/c++/4.4/bits/functexcept.h                             |
```

```
55 | /usr/include/c++/4.4/bits/istream.tcc                                  |
56 | /usr/include/c++/4.4/bits/stl_uninitialized.h                          |
57 | /usr/include/c++/4.4/bits/stringfwd.h                                  |
58 | /usr/include/c++/4.4/bits/stl_iterator_base_types.h                    |
59 | /usr/include/c++/4.4/bits/localefwd.h                                  |
60 | /usr/include/c++/4.4/x86_64-linux-gnu/bits/c++locale.h                 |
61 | /usr/include/c++/4.4/bits/stl_algobase.h                               |
62 | /usr/include/c++/4.4/ostream                                           |
63 | /usr/include/c++/4.4/bits/cpp_type_traits.h                            |
64 | /usr/include/c++/4.4/bits/locale_classes.tcc                           |
65 | /usr/include/c++/4.4/bits/postypes.h                                   |
66 | /usr/include/c++/4.4/bits/stl_vector.h                                 |
67 | /usr/include/c++/4.4/bits/stl_construct.h                              |
68 | /usr/include/c++/4.4/cstdlib                                           |
69 | /usr/include/c++/4.4/bits/stl_tree.h                                   |
70 | /usr/include/c++/4.4/bits/allocator.h                                  |
71 | /usr/include/c++/4.4/streambuf                                         |
72 | /usr/include/c++/4.4/bits/char_traits.h                                |
73 | /usr/include/c++/4.4/bits/basic_string.h                               |
74 | /usr/include/c++/4.4/bits/basic_string.tcc                             |
75 | /usr/include/c++/4.4/x86_64-linux-gnu/bits/ctype_base.h                |
76 | /usr/include/c++/4.4/bits/stl_iterator.h                               |
77 | /usr/include/c++/4.4/bits/ostream_insert.h                             |
78 | /usr/include/c++/4.4/bits/locale_facets.tcc                            |
79 | /usr/include/c++/4.4/bits/ostream.tcc                                  |
80 | /usr/include/c++/4.4/istream                                           |
81 | /usr/include/c++/4.4/bits/streambuf.tcc                                |
82 | <built-in>                                                             |
83 | /usr/include/c++/4.4/ext/numeric_traits.h                              |
84 | /usr/include/c++/4.4/ext/atomicity.h                                   |
85 | /usr/include/c++/4.4/ext/type_traits.h                                 |
86 | /usr/include/c++/4.4/ext/new_allocator.h                               |
87 | /usr/include/c++/4.4/sstream                                           |
88 | /usr/include/c++/4.4/bits/sstream.tcc                                  |
89 | /usr/include/c++/4.4/bits/vector.tcc                                   |
90 | /usr/include/c++/4.4/x86_64-linux-gnu/bits/ctype_inline.h              |
91 | /usr/include/c++/4.4/bits/basic_ios.h                                  |
92 | /usr/include/c++/4.4/bits/basic_ios.tcc                                |
93 | /usr/include/c++/4.4/bits/stl_pair.h                                   |
94 | /usr/include/c++/4.4/bits/stl_function.h                               |
95 | /usr/include/c++/4.4/bits/stl_map.h                                    |
96 | /usr/include/errno.h                                                   |
97 | /usr/include/bits/errno.h                                              |
98 | /usr/include/c++/4.4/backward/auto_ptr.h                               |
99 | /usr/include/c++/4.4/bits/stl_set.h                                    |
```

```
| /usr/include/qt3/qframe.h                                    |
| /usr/include/qt3/qglobal.h                                   |
| /usr/include/qt3/qtextstream.h                               |
| /usr/include/qt3/qmenudata.h                                 |
| /usr/include/qt3/qwindowdefs.h                               |
| /usr/include/qt3/qdict.h                                     |
| /usr/include/qt3/qscrollview.h                               |
| /usr/include/qt3/qpushbutton.h                               |
| /usr/include/qt3/qpixmap.h                                   |
| /usr/include/qt3/qpopupmenu.h                                |
| /usr/include/qt3/qpalette.h                                  |
| /usr/include/qt3/qlistview.h                                 |
| /usr/include/qt3/qcstring.h                                  |
| /usr/include/qt3/qobjectdefs.h                               |
| /usr/include/qt3/qiconset.h                                  |
| /usr/include/qt3/qptrcollection.h                            |
| /usr/include/qt3/qregion.h                                   |
| /usr/include/qt3/qevent.h                                    |
| /usr/include/qt3/qmultilineedit.h                            |
| /usr/include/qt3/qfontmetrics.h                              |
| /usr/include/qt3/qstrlist.h                                  |
| /usr/include/qt3/qvaluelist.h                                |
| /usr/include/qt3/qdockwindow.h                               |
| /usr/include/qt3/qmap.h                                      |
| /usr/include/qt3/qcolor.h                                    |
| /usr/include/qt3/qscrollbar.h                                |
| /usr/include/qt3/qrect.h                                     |
| /usr/include/qt3/qsize.h                                     |
| /usr/include/qt3/qpoint.h                                    |
| /usr/include/qt3/qstring.h                                   |
| /usr/include/qt3/qkeysequence.h                              |
| /usr/include/qt3/qvariant.h                                  |
| /usr/include/qt3/qpaintdevice.h                              |
| /usr/include/qt3/qwidget.h                                   |
| /usr/include/qt3/qobject.h                                   |
| /usr/include/qt3/qgarray.h                                   |
| /usr/include/qt3/qtextedit.h                                 |
| /usr/include/qt3/qgvector.h                                  |
| /usr/include/qt3/qmainwindow.h                               |
| /usr/include/qt3/qfont.h                                     |
| /usr/include/qt3/qbrush.h                                    |
| /usr/include/qt3/qstringlist.h                               |
| /usr/include/qt3/qglist.h                                    |
| /usr/include/qt3/qbutton.h                                   |
| /usr/include/qt3/qptrlist.h                                  |
```

```
| /usr/include/qt3/qfontinfo.h                                                |
| /usr/include/qt3/qmime.h                                                    |
| /usr/include/qt3/qgdict.h                                                   |
| /usr/include/qt3/qdatastream.h                                              |
| /usr/include/qt3/qnamespace.h                                               |
| /usr/include/qt3/qpair.h                                                    |
| /usr/include/qt3/qrangecontrol.h                                            |
| /usr/include/qt3/qmemarray.h                                                |
| /usr/include/qt3/qshared.h                                                  |
| /usr/include/qt3/qmenubar.h                                                 |
| /usr/include/qt3/qptrvector.h                                               |
| /usr/include/qt3/qsizepolicy.h                                              |
| /usr/include/qt3/qtoolbar.h                                                 |
| /usr/include/qt3/qiodevice.h                                                |
| /usr/include/qt3/qstylesheet.h                                              |
| /usr/include/qt3/qlabel.h                                                   |
| /usr/include/qt3/qsignal.h                                                  |
| /usr/include/c++/4.4/bits/stl_list.h                                        |
| /usr/include/c++/4.4/bits/list.tcc                                          |
| /usr/include/qt3/qlayout.h                                                  |
| /usr/include/qt3/qdialog.h                                                  |
| /usr/include/qt3/qapplication.h                                             |
| /usr/include/qt3/qtranslator.h                                              |
| /usr/include/qt3/qasciidict.h                                               |
| /usr/include/qt3/qdesktopwidget.h                                           |
| /usr/include/qt3/qsemimodal.h                                               |
| /usr/include/qt3/qlineedit.h                                                |
| /usr/include/qt3/qcombobox.h                                                |
| /usr/include/c++/4.4/stdexcept                                              |
+-----------------------------------------------------------------------------+
170 rows in set (1.21 sec)
```

for the log4cxx project the output is

The List of Included Library Files for log4cxx

```
+-----------------------------------------------------------------------------+
| name                                                                        |
+-----------------------------------------------------------------------------+
| /usr/include/time.h                                                         |
| /opt/projects/gccxml/install/share/gccxml-0.9/GCC/4.4/gccxml_builtins.h |
| /usr/include/unistd.h                                                       |
| /usr/include/pthread.h                                                      |
| /usr/include/wctype.h                                                       |
| /usr/include/wchar.h                                                        |
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/gthr-default.h                |
```

```
| /usr/include/sched.h                                                    |
| /usr/include/c++/4.4/bits/locale_facets.h                               |
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/atomic_word.h                |
| /usr/include/bits/pthreadtypes.h                                        |
| /usr/include/c++/4.4/iosfwd                                             |
| /usr/include/bits/types.h                                               |
| /usr/include/ctype.h                                                    |
| /usr/include/c++/4.4/new                                                |
| /usr/include/bits/sigset.h                                              |
| /usr/include/signal.h                                                   |
| /usr/include/c++/4.4/exception                                          |
| /usr/include/stdio.h                                                    |
| /usr/include/c++/4.4/bits/locale_classes.h                              |
| /usr/include/bits/confname.h                                            |
| /usr/include/xlocale.h                                                  |
| /usr/include/c++/4.4/cxxabi-forced.h                                    |
| /usr/include/bits/sched.h                                               |
| /usr/include/getopt.h                                                   |
| /usr/include/locale.h                                                   |
| /usr/include/bits/setjmp.h                                              |
| /usr/include/c++/4.4/bits/ios_base.h                                    |
| /usr/lib/gcc/x86_64-linux-gnu/4.4.5/include/stddef.h                    |
| /usr/lib/gcc/x86_64-linux-gnu/4.4.5/include/stdarg.h                    |
| /usr/include/bits/locale.h                                              |
| /usr/include/c++/4.4/bits/stl_bvector.h                                 |
| /usr/include/c++/4.4/bits/functexcept.h                                 |
| /usr/include/c++/4.4/bits/istream.tcc                                   |
| /usr/include/c++/4.4/bits/stl_uninitialized.h                           |
| /usr/include/c++/4.4/bits/stringfwd.h                                   |
| /usr/include/c++/4.4/bits/stl_pair.h                                    |
| /usr/include/c++/4.4/bits/stl_iterator_base_types.h                     |
| /usr/include/c++/4.4/bits/localefwd.h                                   |
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/c++locale.h                  |
| /usr/include/c++/4.4/bits/stl_algobase.h                                |
| /usr/include/c++/4.4/ostream                                            |
| /usr/include/c++/4.4/bits/cpp_type_traits.h                             |
| /usr/include/c++/4.4/bits/stl_vector.h                                  |
| /usr/include/c++/4.4/bits/locale_classes.tcc                            |
| /usr/include/c++/4.4/bits/postypes.h                                    |
| /usr/include/c++/4.4/bits/stl_construct.h                               |
| /usr/include/c++/4.4/bits/stl_tree.h                                    |
| /usr/include/c++/4.4/bits/allocator.h                                   |
| /usr/include/c++/4.4/streambuf                                          |
| /usr/include/c++/4.4/bits/char_traits.h                                 |
| /usr/include/c++/4.4/bits/stl_map.h                                     |
```

```
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/ctype_base.h                |
| /usr/include/c++/4.4/bits/stl_iterator.h                               |
| /usr/include/c++/4.4/bits/ostream_insert.h                             |
| /usr/include/c++/4.4/bits/locale_facets.tcc                            |
| /usr/include/c++/4.4/bits/basic_string.h                               |
| /usr/include/c++/4.4/bits/streambuf.tcc                                |
| /usr/include/c++/4.4/bits/ostream.tcc                                  |
| /usr/include/c++/4.4/istream                                           |
| /usr/include/c++/4.4/bits/basic_string.tcc                             |
| /usr/include/c++/4.4/bits/stl_function.h                               |
| <built-in>                                                             |
| /usr/include/c++/4.4/ext/numeric_traits.h                              |
| /usr/include/c++/4.4/ext/atomicity.h                                   |
| /usr/include/c++/4.4/ext/type_traits.h                                 |
| /usr/include/c++/4.4/ext/new_allocator.h                               |
| /usr/include/c++/4.4/sstream                                           |
| /usr/include/c++/4.4/bits/vector.tcc                                   |
| /usr/include/c++/4.4/bits/sstream.tcc                                  |
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/ctype_inline.h              |
| /usr/include/c++/4.4/bits/basic_ios.h                                  |
| /usr/include/c++/4.4/bits/basic_ios.tcc                                |
| /usr/include/c++/4.4/bits/stl_list.h                                   |
| /usr/include/c++/4.4/bits/list.tcc                                     |
| /usr/include/c++/4.4/bits/stl_deque.h                                  |
| /usr/include/c++/4.4/bits/stl_stack.h                                  |
| /usr/include/c++/4.4/bits/deque.tcc                                    |
| /usr/include/libio.h                                                   |
| /usr/include/_G_config.h                                               |
| /usr/include/bits/sys_errlist.h                                        |
| /usr/include/string.h                                                  |
| /usr/include/apr-1.0/apr.h                                             |
| /usr/include/bits/time.h                                               |
| /usr/include/sys/socket.h                                              |
| /usr/include/stdint.h                                                  |
| /usr/include/apr-1.0/apr_allocator.h                                   |
| /usr/include/apr-1.0/apr_pools.h                                       |
| /usr/include/sys/ucontext.h                                            |
| /usr/include/sys/resource.h                                            |
| /usr/include/bits/resource.h                                           |
| /usr/include/bits/siginfo.h                                            |
| /usr/include/bits/sigcontext.h                                         |
| /usr/include/sys/types.h                                               |
| /usr/include/apr-1.0/apr_general.h                                     |
| /usr/include/apr-1.0/apr_strings.h                                     |
| /usr/include/sys/uio.h                                                 |
```

```
| /usr/include/sys/select.h                                    |
| /usr/include/errno.h                                         |
| /usr/include/apr-1.0/apr_thread_mutex.h                      |
| /usr/include/bits/sigthread.h                                |
| /usr/include/bits/uio.h                                      |
| /usr/include/sys/sysmacros.h                                 |
| /usr/include/bits/sigstack.h                                 |
| /usr/include/bits/socket.h                                   |
| /usr/include/apr-1.0/apr_errno.h                             |
| /usr/include/bits/sigaction.h                                |
| /usr/include/sys/wait.h                                      |
| /usr/include/bits/errno.h                                    |
| /usr/include/bits/waitstatus.h                               |
| /usr/include/bits/sockaddr.h                                 |
| /usr/include/apr-1.0/apr_file_info.h                         |
| /usr/include/apr-1.0/apr_file_io.h                           |
| /usr/include/apr-1.0/apr_time.h                              |
| /usr/include/apr-1.0/apr_tables.h                            |
| /usr/include/apr-1.0/apr_user.h                              |
| /usr/include/assert.h                                        |
| /usr/include/apr-1.0/apr_atomic.h                            |
| /usr/include/apr-1.0/apr_thread_proc.h                       |
| /usr/include/stdlib.h                                        |
| /usr/include/sys/time.h                                      |
| /usr/include/alloca.h                                        |
| /usr/include/c++/4.4/bits/stl_algo.h                         |
| /usr/include/c++/4.4/cstdlib                                 |
| /usr/include/apr-1.0/apr_signal.h                            |
| /usr/include/apr-1.0/apr_network_io.h                        |
| /usr/include/netinet/in.h                                    |
| /usr/include/bits/in.h                                       |
| /usr/include/sys/stat.h                                      |
| /usr/include/apr-1.0/apr_portable.h                          |
| /usr/include/apr-1.0/apr_xlate.h                             |
| /usr/include/apr-1.0/apr_shm.h                               |
| /usr/include/apr-1.0/apr_dso.h                               |
| /usr/include/bits/stat.h                                     |
| /usr/include/dirent.h                                        |
| /usr/include/apr-1.0/apr_global_mutex.h                      |
| /usr/include/bits/fcntl.h                                    |
| /usr/include/apr-1.0/apr_proc_mutex.h                        |
| /usr/include/fcntl.h                                         |
| /usr/include/bits/dirent.h                                   |
| /usr/include/apr-1.0/apr_lib.h                               |
| /usr/include/apr-1.0/apr_thread_cond.h                       |
```

```
| /usr/include/apr-1.0/apr_env.h                                              |
| /usr/include/c++/4.4/stdexcept                                              |
| /usr/include/c++/4.4/limits                                                 |
| /usr/include/sys/syslog.h                                                   |
| /usr/include/c++/4.4/bits/locale_facets_nonio.h                             |
| /usr/include/c++/4.4/bits/codecvt.h                                         |
| /usr/include/libintl.h                                                      |
| /usr/include/c++/4.4/bits/locale_facets_nonio.tcc                           |
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/messages_members.h               |
| /usr/include/c++/4.4/bits/streambuf_iterator.h                              |
| /usr/include/c++/4.4/iostream                                               |
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/basic_file.h                     |
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/c++io.h                          |
| /usr/include/c++/4.4/fstream                                                |
| /usr/include/c++/4.4/bits/fstream.tcc                                       |
| /usr/include/apr-1.0/apr_xml.h                                              |
| /usr/include/apr-1.0/apr_poll.h                                             |
+-----------------------------------------------------------------------------+
159 rows in set (1 min 0.30 sec)
```

and for the C++ Sockets Library the output is

The List of Included Library Files for the C++ Sockets Library

```
+-----------------------------------------------------------------------------+
| name                                                                        |
+-----------------------------------------------------------------------------+
| /opt/projects/gccxml/install/share/gccxml-0.9/GCC/4.4/gccxml_builtins.h |
| /usr/include/bits/time.h                                                    |
| /usr/include/stdint.h                                                       |
| /usr/include/bits/types.h                                                   |
| /usr/include/bits/sigset.h                                                  |
| /usr/include/sys/select.h                                                   |
| /usr/include/sys/time.h                                                     |
| /usr/include/time.h                                                         |
| /usr/include/inttypes.h                                                     |
| /usr/include/string.h                                                       |
| /usr/include/stdio.h                                                        |
| /usr/include/pthread.h                                                      |
| /usr/include/unistd.h                                                       |
| /usr/include/wchar.h                                                        |
| /usr/include/sys/socket.h                                                   |
| /usr/include/stdlib.h                                                       |
| /usr/include/netinet/in.h                                                   |
| /usr/include/sched.h                                                        |
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/atomic_word.h                    |
```

```
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/gthr-default.h |
| /usr/include/bits/pthreadtypes.h |
| /usr/include/ctype.h |
| /usr/include/c++/4.4/new |
| /usr/include/libio.h |
| /usr/include/sys/types.h |
| /usr/include/c++/4.4/exception |
| /usr/include/sys/uio.h |
| /usr/include/_G_config.h |
| /usr/include/bits/setjmp.h |
| /usr/include/bits/in.h |
| /usr/include/bits/socket.h |
| /usr/include/arpa/inet.h |
| /usr/include/bits/uio.h |
| /usr/include/xlocale.h |
| /usr/include/c++/4.4/bits/stl_list.h |
| /usr/include/c++/4.4/cxxabi-forced.h |
| /usr/include/sys/sysmacros.h |
| /usr/include/bits/sched.h |
| /usr/include/bits/sys_errlist.h |
| /usr/include/locale.h |
| /usr/include/getopt.h |
| /usr/include/alloca.h |
| /usr/lib/gcc/x86_64-linux-gnu/4.4.5/include/stddef.h |
| /usr/include/bits/confname.h |
| /usr/include/bits/waitstatus.h |
| /usr/lib/gcc/x86_64-linux-gnu/4.4.5/include/stdarg.h |
| /usr/include/bits/locale.h |
| /usr/include/bits/sockaddr.h |
| /usr/include/c++/4.4/bits/functexcept.h |
| /usr/include/c++/4.4/bits/stl_uninitialized.h |
| /usr/include/c++/4.4/iosfwd |
| /usr/include/c++/4.4/bits/stringfwd.h |
| /usr/include/c++/4.4/bits/stl_pair.h |
| /usr/include/c++/4.4/bits/cpp_type_traits.h |
| /usr/include/c++/4.4/bits/stl_iterator_base_types.h |
| /usr/include/c++/4.4/bits/localefwd.h |
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/c++locale.h |
| /usr/include/c++/4.4/bits/stl_algobase.h |
| /usr/include/c++/4.4/bits/stl_construct.h |
| /usr/include/c++/4.4/bits/stl_tree.h |
| /usr/include/c++/4.4/bits/stl_function.h |
| /usr/include/c++/4.4/bits/postypes.h |
| /usr/include/c++/4.4/bits/allocator.h |
| /usr/include/c++/4.4/bits/char_traits.h |
```

```
| /usr/include/c++/4.4/bits/stl_map.h                                    |
| /usr/include/c++/4.4/backward/auto_ptr.h                               |
| /usr/include/c++/4.4/bits/stl_iterator.h                               |
| /usr/include/c++/4.4/bits/ostream_insert.h                             |
| /usr/include/c++/4.4/bits/basic_string.h                               |
| /usr/include/c++/4.4/bits/basic_string.tcc                             |
| <built-in>                                                             |
| /usr/include/c++/4.4/ext/numeric_traits.h                              |
| /usr/include/c++/4.4/ext/atomicity.h                                   |
| /usr/include/c++/4.4/ext/type_traits.h                                 |
| /usr/include/c++/4.4/ext/new_allocator.h                               |
| /usr/include/signal.h                                                  |
| /usr/include/c++/4.4/bits/stl_bvector.h                                |
| /usr/include/c++/4.4/bits/stl_vector.h                                 |
| /usr/include/c++/4.4/bits/list.tcc                                     |
| /usr/include/c++/4.4/bits/vector.tcc                                   |
| /usr/include/openssl/bio.h                                             |
| /usr/include/openssl/asn1.h                                            |
| /usr/include/sys/stat.h                                                |
| /usr/include/openssl/ssl.h                                             |
| /usr/include/openssl/x509.h                                            |
| /usr/include/openssl/ec.h                                              |
| /usr/include/openssl/bn.h                                              |
| /usr/include/openssl/pkcs7.h                                           |
| /usr/include/openssl/ossl_typ.h                                        |
| /usr/include/openssl/evp.h                                             |
| /usr/include/openssl/stack.h                                           |
| /usr/include/openssl/pem.h                                             |
| /usr/include/openssl/tls1.h                                            |
| /usr/include/openssl/pqueue.h                                          |
| /usr/include/openssl/crypto.h                                          |
| /usr/include/openssl/x509_vfy.h                                        |
| /usr/include/openssl/rsa.h                                             |
| /usr/include/openssl/lhash.h                                           |
| /usr/include/semaphore.h                                               |
| /usr/include/openssl/objects.h                                         |
| /usr/include/openssl/dtls1.h                                           |
| /usr/include/openssl/ecdsa.h                                           |
| /usr/include/openssl/dsa.h                                             |
| /usr/include/errno.h                                                   |
| /usr/include/openssl/ssl2.h                                            |
| /usr/include/openssl/dh.h                                              |
| /usr/include/openssl/ecdh.h                                            |
| /usr/include/openssl/sha.h                                             |
| /usr/include/openssl/buffer.h                                          |
```

```
| /usr/include/openssl/ssl3.h                                                |
| /usr/include/openssl/comp.h                                                |
| /usr/include/bits/errno.h                                                  |
| /usr/include/bits/semaphore.h                                              |
| /usr/include/bits/stat.h                                                   |
| /usr/include/openssl/hmac.h                                                |
| /usr/include/netdb.h                                                       |
| /usr/include/rpc/netdb.h                                                   |
| /usr/include/bits/siginfo.h                                                |
| /usr/include/bits/netdb.h                                                  |
| /usr/include/wctype.h                                                      |
| /usr/include/c++/4.4/bits/locale_facets.h                                  |
| /usr/include/c++/4.4/bits/locale_classes.h                                 |
| /usr/include/c++/4.4/bits/ios_base.h                                       |
| /usr/include/c++/4.4/bits/istream.tcc                                      |
| /usr/include/c++/4.4/ostream                                               |
| /usr/include/c++/4.4/bits/locale_classes.tcc                               |
| /usr/include/c++/4.4/streambuf                                             |
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/ctype_base.h                    |
| /usr/include/c++/4.4/bits/locale_facets.tcc                                |
| /usr/include/c++/4.4/bits/streambuf.tcc                                    |
| /usr/include/c++/4.4/bits/ostream.tcc                                      |
| /usr/include/c++/4.4/istream                                               |
| /usr/include/c++/4.4/sstream                                               |
| /usr/include/c++/4.4/bits/sstream.tcc                                      |
| /usr/include/c++/4.4/x86_64-linux-gnu/bits/ctype_inline.h                  |
| /usr/include/c++/4.4/bits/basic_ios.h                                      |
| /usr/include/c++/4.4/bits/basic_ios.tcc                                    |
| /usr/include/fcntl.h                                                       |
| /usr/include/bits/fcntl.h                                                  |
+----------------------------------------------------------------------------+
139 rows in set (10.68 sec)
```

A-1.17 Number of Included Library Files

The output for the libodbc++ project for this query is

Number of Included Library Files for libodbc++

```
+----------------------------------+
| Number of Included Library Files |
+----------------------------------+
|                              170 |
+----------------------------------+
1 row in set (0.97 sec)
```

for the log4cxx project the output is

Number of Included Library Files for log4cxx

```
+----------------------------------+
| Number of Included Library Files |
+----------------------------------+
|                              159 |
+----------------------------------+
1 row in set (1 min 6.33 sec)
```

and for the C++ Sockets Library the output is

Number of Included Library Files for the C++ Sockets Library

```
+----------------------------------+
| Number of Included Library Files |
+----------------------------------+
|                              139 |
+----------------------------------+
1 row in set (10.41 sec)
```

A-1.18 File Weight

The output for the libodbc++ project for this query is

The Individual File Weights for libodbc++

```
+----+-----------------------+-------------+
| id | name                  | File Weight |
+----+-----------------------+-------------+
|  1 | statement.cpp         |           1 |
|  2 | driverinfo.h          |           7 |
|  3 | resultsetmetadata.cpp |           1 |
|  4 | threads.cpp           |           0 |
|  5 | errorhandler.cpp      |           1 |
|  6 | drivermanager.cpp     |           1 |
```

```
|  7 | datastream.cpp         |           0 |
|  8 | callablestatement.cpp  |           1 |
|  9 | dtconv.h               |          10 |
| 10 | datetime.cpp           |           1 |
| 11 | datahandler.h          |           4 |
| 12 | connection.cpp         |           1 |
| 13 | preparedstatement.cpp  |           1 |
| 14 | datahandler.cpp        |           2 |
| 15 | databasemetadata.cpp   |           1 |
| 16 | datastream.h           |           0 |
| 17 | driverinfo.cpp         |           2 |
| 18 | resultset.cpp          |           0 |
| 19 | types.h                |          28 |
| 20 | resultsetmetadata.h    |           6 |
| 21 | config-win32.h         |           0 |
| 22 | statement.h            |          14 |
| 23 | drivermanager.h        |           4 |
| 24 | databasemetadata.h     |           7 |
| 25 | threads.h              |           0 |
| 26 | callablestatement.h    |           3 |
| 27 | errorhandler.h         |          21 |
| 28 | config.h               |           0 |
| 29 | connection.h           |          19 |
| 30 | preparedstatement.h    |           7 |
| 31 | resultset.h            |           9 |
| 32 | setup.h                |           0 |
| 33 | mainwindow.cpp         |           0 |
| 34 | resultwindow.h         |           2 |
| 35 | connectwindow.cpp      |           0 |
| 36 | connectwindow.h        |           2 |
| 37 | resultwindow.cpp       |           2 |
| 38 | main.cpp               |           1 |
| 39 | mainwindow.h           |           3 |
| 40 | isql++.cpp             |           0 |
| 41 | isql++.h               |           2 |
| 42 | config.h               |           0 |
+----+------------------------+-------------+
42 rows in set (0.11 sec)
```

for the log4cxx project the output is

The Individual File Weights for log4cxx

```
+-----+-----------------------------------------+-------------+
| id  | name                                    | File Weight |
+-----+-----------------------------------------+-------------+
```

1	defaultconfigurator.h	3
2	odbcappender.h	3
3	rollingfileappender.h	3
4	ndc.h	9
5	layout.h	56
6	hierarchy.h	3
7	htmllayout.h	3
8	basicconfigurator.h	2
9	file.h	40
10	provisionnode.h	4
11	xmllayout.h	4
12	domconfigurator.h	4
13	levelrangefilter.h	3
14	andfilter.h	1
15	mapfilter.h	1
16	expressionfilter.h	1
17	stringmatchfilter.h	3
18	levelmatchfilter.h	3
19	propertyfilter.h	1
20	denyallfilter.h	2
21	locationinfofilter.h	1
22	loggerfactory.h	174
23	rootlogger.h	4
24	hierarchyeventlistener.h	175
25	defaultrepositoryselector.h	3
26	optionhandler.h	178
27	triggeringeventevaluator.h	6
28	repositoryselector.h	20
29	appenderattachable.h	176
30	locationinfo.h	176
31	errorhandler.h	45
32	filter.h	57
33	loggerrepository.h	174
34	configurator.h	13
35	loggingevent.h	134
36	defaultloggerfactory.h	5
37	filewatchdog.h	4
38	iso8601dateformat.h	7
39	outputstream.h	186
40	stringhelper.h	54
41	dateformat.h	26
42	datelayout.h	5
43	inetaddress.h	26
44	optionconverter.h	27
45	bytearrayoutputstream.h	3

46	appenderattachableimpl.h	175
47	strictmath.h	1
48	cacheddateformat.h	7
49	transform.h	5
50	synchronized.h	23
51	class.h	272
52	locale.h	3
53	objectptr.h	276
54	xml.h	2
55	resourcebundle.h	178
56	reader.h	4
57	aprinitializer.h	13
58	socketoutputstream.h	4
59	strftimedateformat.h	3
60	condition.h	5
61	datagramsocket.h	7
62	simpledateformat.h	11
63	bufferedoutputstream.h	1
64	timezone.h	28
65	absolutetimedateformat.h	3
66	classregistration.h	272
67	properties.h	11
68	bytearrayinputstream.h	2
69	bufferedwriter.h	3
70	onlyonceerrorhandler.h	3
71	propertyresourcebundle.h	3
72	date.h	8
73	system.h	4
74	mutex.h	193
75	stringtokenizer.h	4
76	serversocket.h	6
77	outputstreamwriter.h	19
78	threadspecificdata.h	6
79	fileoutputstream.h	4
80	inputstream.h	20
81	socket.h	16
82	charsetdecoder.h	9
83	thread.h	20
84	systemerrwriter.h	4
85	objectimpl.h	262
86	integer.h	4
87	loglog.h	41
88	object.h	270
89	fileinputstream.h	4
90	transcoder.h	306

91	datagrampacket.h	9
92	cyclicbuffer.h	6
93	relativetimedateformat.h	3
94	datetimedateformat.h	3
95	syslogwriter.h	5
96	tchar.h	0
97	inputstreamreader.h	2
98	messagebuffer.h	176
99	loader.h	5
100	charsetencoder.h	183
101	bytebuffer.h	17
102	systemoutwriter.h	3
103	exception.h	100
104	objectoutputstream.h	178
105	pool.h	221
106	threadlocal.h	3
107	writer.h	29
108	log4cxx.h	317
109	rollingpolicy.h	19
110	rollingfileappender.h	5
111	rolloverdescription.h	21
112	filterbasedtriggeringpolicy.h	4
113	rollingpolicybase.h	10
114	action.h	29
115	timebasedrollingpolicy.h	4
116	triggeringpolicy.h	21
117	rollingfileappenderskeleton.h	9
118	manualtriggeringpolicy.h	4
119	filerenameaction.h	4
120	sizebasedtriggeringpolicy.h	4
121	gzcompressaction.h	4
122	zipcompressaction.h	4
123	fixedwindowrollingpolicy.h	5
124	log4cxx_private.h	0
125	methodlocationpatternconverter.h	3
126	patternparser.h	16
127	ndcpatternconverter.h	3
128	filedatepatternconverter.h	4
129	loggerpatternconverter.h	3
130	formattinginfo.h	18
131	fulllocationpatternconverter.h	3
132	loggingeventpatternconverter.h	44
133	threadpatternconverter.h	3
134	linelocationpatternconverter.h	3
135	levelpatternconverter.h	3

136	throwableinformationpatternconverter.h	3
137	lineseparatorpatternconverter.h	3
138	datepatternconverter.h	5
139	literalpatternconverter.h	4
140	namepatternconverter.h	7
141	nameabbreviator.h	9
142	classnamepatternconverter.h	3
143	filelocationpatternconverter.h	3
144	relativetimepatternconverter.h	3
145	patternconverter.h	58
146	propertiespatternconverter.h	3
147	integerpatternconverter.h	5
148	messagepatternconverter.h	3
149	asyncappender.h	3
150	portability.h	0
151	dailyrollingfileappender.h	3
152	logmanager.h	16
153	appender.h	177
154	fileappender.h	11
155	consoleappender.h	4
156	propertysetter.h	7
157	nteventlogappender.h	1
158	outputdebugstringappender.h	1
159	writerappender.h	16
160	socketappender.h	3
161	syslogappender.h	3
162	telnetappender.h	3
163	socketappenderskeleton.h	7
164	sockethubappender.h	3
165	smtpappender.h	4
166	xmlsocketappender.h	3
167	logstring.h	306
168	simplelayout.h	3
169	level.h	176
170	fallbackerrorhandler.h	2
171	appenderskeleton.h	39
172	mdc.h	184
173	ttcclayout.h	3
174	propertyconfigurator.h	4
175	stream.h	2
176	logger.h	174
177	patternlayout.h	5
178	integer.cpp	1
179	locationinfo.cpp	1
180	properties.cpp	1

181	file.cpp	1
182	inputstreamreader.cpp	0
183	hierarchy.cpp	1
184	socket.cpp	1
185	socketappender.cpp	1
186	aprinitializer.cpp	1
187	propertiespatternconverter.cpp	1
188	ndc.cpp	1
189	threadpatternconverter.cpp	1
190	basicconfigurator.cpp	1
191	fulllocationpatternconverter.cpp	1
192	propertyconfigurator.cpp	1
193	nameabbreviator.cpp	1
194	objectoutputstream.cpp	1
195	outputstream.cpp	1
196	socketappenderskeleton.cpp	1
197	logstream.cpp	1
198	resourcebundle.cpp	1
199	relativetimepatternconverter.cpp	1
200	charsetencoder.cpp	1
201	inetaddress.cpp	1
202	throwableinformationpatternconverter.cpp	1
203	patternconverter.cpp	1
204	exception.cpp	1
205	rollingpolicybase.cpp	1
206	classregistration.cpp	1
207	stringmatchfilter.cpp	1
208	datagrampacket.cpp	1
209	datagramsocket.cpp	1
210	layout.cpp	1
211	threadlocal.cpp	1
212	levelmatchfilter.cpp	1
213	htmllayout.cpp	1
214	fallbackerrorhandler.cpp	1
215	writer.cpp	1
216	charsetdecoder.cpp	1
217	obsoleterollingfileappender.cpp	1
218	triggeringpolicy.cpp	1
219	fileoutputstream.cpp	1
220	patternparser.cpp	1
221	integerpatternconverter.cpp	1
222	systemerrwriter.cpp	1
223	transform.cpp	1
224	sizebasedtriggeringpolicy.cpp	1
225	onlyonceerrorhandler.cpp	1

226	outputstreamwriter.cpp	1
227	writerappender.cpp	1
228	datepatternconverter.cpp	1
229	class.cpp	1
230	telnetappender.cpp	1
231	rolloverdescription.cpp	1
232	nteventlogappender.cpp	0
233	dateformat.cpp	1
234	filterbasedtriggeringpolicy.cpp	1
235	objectptr.cpp	1
236	timezone.cpp	1
237	systemoutwriter.cpp	1
238	patternlayout.cpp	1
239	asyncappender.cpp	1
240	objectimpl.cpp	1
241	filedatepatternconverter.cpp	1
242	sockethubappender.cpp	1
243	filerenameaction.cpp	1
244	fileinputstream.cpp	1
245	zipcompressaction.cpp	1
246	system.cpp	1
247	logmanager.cpp	1
248	fixedwindowrollingpolicy.cpp	1
249	stringhelper.cpp	1
250	filewatchdog.cpp	1
251	cacheddateformat.cpp	1
252	appenderattachableimpl.cpp	1
253	reader.cpp	1
254	formattinginfo.cpp	1
255	odbcappender.cpp	1
256	methodlocationpatternconverter.cpp	1
257	bufferedwriter.cpp	1
258	bytearrayinputstream.cpp	1
259	strftimedateformat.cpp	1
260	syslogappender.cpp	1
261	rollingpolicy.cpp	1
262	rollingfileappender.cpp	1
263	loggingevent.cpp	1
264	gzcompressaction.cpp	1
265	namepatternconverter.cpp	1
266	outputdebugstringappender.cpp	0
267	loggingeventpatternconverter.cpp	1
268	simpledateformat.cpp	1
269	filelocationpatternconverter.cpp	1
270	threadspecificdata.cpp	1

271	xmlsocketappender.cpp	1
272	inputstream.cpp	1
273	optionconverter.cpp	1
274	appenderskeleton.cpp	1
275	date.cpp	1
276	locale.cpp	1
277	propertysetter.cpp	1
278	synchronized.cpp	1
279	propertyresourcebundle.cpp	1
280	bytebuffer.cpp	1
281	literalpatternconverter.cpp	1
282	simplelayout.cpp	1
283	loggerpatternconverter.cpp	1
284	levelrangefilter.cpp	1
285	configurator.cpp	1
286	ttcclayout.cpp	1
287	syslogwriter.cpp	1
288	manualtriggeringpolicy.cpp	1
289	messagebuffer.cpp	1
290	level.cpp	1
291	threadcxx.cpp	1
292	cyclicbuffer.cpp	1
293	loglog.cpp	1
294	fileappender.cpp	1
295	mutex.cpp	1
296	stringtokenizer.cpp	1
297	filter.cpp	1
298	datelayout.cpp	1
299	action.cpp	1
300	defaultloggerfactory.cpp	1
301	loader.cpp	1
302	domconfigurator.cpp	1
303	levelpatternconverter.cpp	1
304	relativetimedateformat.cpp	1
305	serversocket.cpp	1
306	timebasedrollingpolicy.cpp	1
307	defaultconfigurator.cpp	1
308	smtpappender.cpp	1
309	rootlogger.cpp	1
310	pool.cpp	1
311	socketoutputstream.cpp	0
312	messagepatternconverter.cpp	1
313	transcoder.cpp	1
314	dailyrollingfileappender.cpp	1
315	defaultrepositoryselector.cpp	1

```
| 316 | condition.cpp                              |           1 |
| 317 | lineseparatorpatternconverter.cpp          |           1 |
| 318 | classnamepatternconverter.cpp              |           1 |
| 319 | logger.cpp                                 |           1 |
| 320 | mdc.cpp                                    |           1 |
| 321 | xmllayout.cpp                              |           1 |
| 322 | ndcpatternconverter.cpp                    |           1 |
| 323 | bytearrayoutputstream.cpp                  |           1 |
| 324 | consoleappender.cpp                        |           1 |
| 325 | linelocationpatternconverter.cpp           |           1 |
+-----+--------------------------------------------+-------------+
325 rows in set (10.57 sec)
```

and for the C++ Sockets Library the output is

The Individual File Weights for the C++ Sockets Library

```
+-----+--------------------------+-------------+
| id  | name                     | File Weight |
+-----+--------------------------+-------------+
|   1 | Event.h                  |           4 |
|   2 | Debug.cpp                |           0 |
|   3 | IHttpServer.h            |           5 |
|   4 | SocketStream.cpp         |           0 |
|   5 | HttpRequest.h            |           6 |
|   6 | EventTime.h              |           6 |
|   7 | HttpRequest.cpp          |           2 |
|   8 | HttpPostSocket.h         |           2 |
|   9 | Thread.cpp               |           0 |
|  10 | HttpBaseSocket.h         |           2 |
|  11 | MemFile.cpp              |           0 |
|  12 | Ajp13Socket.cpp          |           0 |
|  13 | HttpPostSocket.cpp       |           2 |
|  14 | XmlException.cpp         |           0 |
|  15 | SocketThread.cpp         |           0 |
|  16 | EventHandler.h           |           2 |
|  17 | Exception.h              |          13 |
|  18 | IEventOwner.h            |           4 |
|  19 | Lock.h                   |           9 |
|  20 | Sockets-config.cpp       |           0 |
|  21 | Ipv4Address.h            |           7 |
|  22 | ResolvServer.cpp         |           0 |
|  23 | StreamSocket.cpp         |           0 |
|  24 | SctpSocket.h             |           0 |
|  25 | HttpResponse.cpp         |           0 |
|  26 | EventHandler.cpp         |           2 |
```

	27	SocketStream.h	2
	28	SocketHandlerEp.cpp	0
	29	UdpSocket.cpp	0
	30	Mutex.cpp	0
	31	UdpSocket.h	4
	32	StdoutLog.cpp	0
	33	ResolvSocket.cpp	0
	34	StreamWriter.h	2
	35	HttpdCookies.cpp	0
	36	FileStream.h	2
	37	File.cpp	0
	38	Utility.cpp	0
	39	XmlDocument.h	0
	40	sockets-config.h	0
	41	EventTime.cpp	2
	42	HTTPSocket.cpp	0
	43	SctpSocket.cpp	0
	44	SocketThread.h	4
	45	socket_include.cpp	0
	46	Parse.cpp	0
	47	IFileUpload.h	3
	48	AjpBaseSocket.h	4
	49	Debug.h	2
	50	HttpdForm.h	6
	51	SocketHandlerThread.h	4
	52	SSLInitializer.cpp	0
	53	Ipv6Address.h	0
	54	SocketHandlerThread.cpp	2
	55	SmtpdSocket.cpp	0
	56	XmlNode.h	0
	57	Thread.h	50
	58	Base64.cpp	0
	59	SocketHandler.h	10
	60	Lock.cpp	2
	61	XmlException.h	0
	62	HttpTransaction.cpp	0
	63	FileStream.cpp	2
	64	Base64.h	4
	65	TcpSocket.cpp	0
	66	StreamWriter.cpp	2
	67	SocketHandler.cpp	2
	68	Json.h	2
	69	Mutex.h	40
	70	Exception.cpp	2
	71	IStream.h	9

72	HttpBaseSocket.cpp	2
73	HTTPSocket.h	16
74	IFile.h	23
75	ListenSocket.h	0
76	ResolvSocket.h	4
77	HttpResponse.h	6
78	Semaphore.h	52
79	IEventOwner.cpp	2
80	IBase.h	1
81	TcpSocket.h	30
82	Semaphore.cpp	2
83	XmlDocument.cpp	0
84	HttpDebugSocket.h	2
85	AjpBaseSocket.cpp	2
86	SSLInitializer.h	32
87	HttpDebugSocket.cpp	2
88	socket_include.h	75
89	SocketAddress.h	49
90	StdLog.h	29
91	HttpdSocket.h	2
92	HttpPutSocket.h	2
93	ISocketHandler.h	26
94	HttpdCookies.h	10
95	ResolvServer.h	4
96	Ipv6Address.cpp	0
97	XmlNode.cpp	0
98	MemFile.h	6
99	SmtpdSocket.h	2
100	HttpGetSocket.h	2
101	HttpClientSocket.h	8
102	Ajp13Socket.h	2
103	Event.cpp	2
104	HttpPutSocket.cpp	2
105	HttpdSocket.cpp	2
106	HttpTransaction.h	10
107	Ipv4Address.cpp	2
108	IEventHandler.h	5
109	Socket.h	46
110	HttpClientSocket.cpp	2
111	IMutex.h	43
112	Parse.h	6
113	Json.cpp	2
114	Socket.cpp	2
115	StreamSocket.h	35
116	StdoutLog.h	2

```
| 117 | HttpdForm.cpp           |            2 |
| 118 | File.h                  |           16 |
| 119 | ajp13.h                 |            0 |
| 120 | Utility.h               |           47 |
| 121 | SocketHandlerEp.h       |            2 |
| 122 | HttpGetSocket.cpp       |            2 |
+-----+-------------------------+--------------+
122 rows in set (0.34 sec)
```

A-2 Composite Query Output

A-2.1 Average SLOC for all Files

The output for the libodbc++ project for this query is

The Average SLOC for All Files in libodbc++

```
+---------------+
| Average SLOC |
+---------------+
|     391.5238 |
+---------------+
1 row in set (0.11 sec)
```

for the log4cxx project the output is

The Average SLOC for All Files in log4cxx

```
+---------------+
| Average SLOC |
+---------------+
|     140.7169 |
+---------------+
1 row in set (0.00 sec)
```

and for the C++ Sockets Library the output is

The Average SLOC for All Files in the C++ Sockets Library

```
+--------------+
| Average SLOC |
+--------------+
|     220.2377 |
+--------------+
1 row in set (0.12 sec)
```

A-2.2 Average SLOC for Header Files

The output for the libodbc++ project for this query is

Average SLOC for Header Files in libodbc++

```
+------------------+
| Average .h SLOC |
+------------------+
|         278.6087 |
+------------------+
1 row in set (0.04 sec)
```

for the log4cxx project the output is

Average SLOC for Header Files in log4cxx

```
+------------------+
| Average .h SLOC |
+------------------+
|         126.3051 |
+------------------+
1 row in set (0.00 sec)
```

and for the C++ Sockets Library the output is

Average SLOC for Header Files in the C++ Sockets Library

```
+------------------+
| Average .h SLOC |
+------------------+
|         124.0299 |
+------------------+
1 row in set (0.02 sec)
```

A-2.3 Average SLOC for Source Files

The output for the libodbc++ project for this query is

Average SLOC for Source Files in libodbc++

```
+--------------------+
| Average .cpp SLOC |
+--------------------+
|           528.2105 |
+--------------------+
1 row in set (0.07 sec)
```

for the log4cxx project the output is

Average SLOC for Source Files in log4cxx

```
+--------------------+
| Average .cpp SLOC |
+--------------------+
|           157.9527 |
+--------------------+
1 row in set (0.00 sec)
```

and for the C++ Sockets Library the output is

Average SLOC for Source Files in the C++ Sockets Library

```
+--------------------+
| Average .cpp SLOC |
+--------------------+
|           337.4364 |
+--------------------+
1 row in set (0.09 sec)
```

A-2.4 Proportion of Private, Protected and Public Variables

The output for the libodbc++ project for this query is

Proportion of Variables for libodbc++

```
+--------------------------------+
| Private to Protected to Public |
+--------------------------------+
| 81 to 0 to 311                 |
+--------------------------------+
1 row in set (0.09 sec)
```

for the log4cxx project the output is

Proportion of Variables for log4cxx

```
+--------------------------------+
| Private to Protected to Public |
+--------------------------------+
| 23 to 0 to 1029                |
+--------------------------------+
1 row in set (0.92 sec)
```

and for the C++ Sockets Library the output is

Proportion of Variables for the C++ Sockets Library

```
+--------------------------------+
| Private to Protected to Public |
+--------------------------------+
| 40 to 1 to 244                 |
+--------------------------------+
1 row in set (0.05 sec)
```

A-2.5 Percentage Private Variables

The output for the libodbc++ project for this query is

Percentage Private Variables for libodbc++

```
+------------------------------+
| Percentage Private Variables |
+------------------------------+
| 20.6633%                     |
+------------------------------+
1 row in set (0.08 sec)
```

for the log4cxx project the output is

Percentage Private Variables for log4cxx

```
+------------------------------+
| Percentage Private Variables |
+------------------------------+
| 2.1905%                      |
+------------------------------+
1 row in set (0.81 sec)
```

and for the C++ Sockets Library the output is

Percentage Private Variables for the C++ Sockets Library

```
+------------------------------+
| Percentage Private Variables |
+------------------------------+
| 14.0351%                     |
+------------------------------+
1 row in set (0.05 sec)
```

A-2.6 Percentage Protected Variables

The output for the libodbc++ project for this query is

Percentage Protected Variables for libodbc++

```
+--------------------------------+
| Percentage Protected Variables |
+--------------------------------+
| 0.0000%                        |
+--------------------------------+
1 row in set (0.08 sec)
```

for the log4cxx project the output is

Percentage Protected Variables for log4cxx

```
+--------------------------------+
| Percentage Protected Variables |
+--------------------------------+
| 0.0000%                        |
+--------------------------------+
1 row in set (0.82 sec)
```

and for the C++ Sockets Library the output is

Percentage Protected Variables for the C++ Sockets Library

```
+-------------------------------+
| Percentage Protected Variables |
+-------------------------------+
| 0.3509%                       |
+-------------------------------+
1 row in set (0.11 sec)
```

A-2.7 Percentage Public Variables

The output for the libodbc++ project for this query is

Percentage Public Variables for libodbc++

```
+-----------------------------+
| Percentage Public Variables |
+-----------------------------+
| 79.3367%                    |
+-----------------------------+
1 row in set (0.13 sec)
```

for the log4cxx project the output is

Percentage Public Variables for log4cxx

```
+-----------------------------+
| Percentage Public Variables |
+-----------------------------+
| 98.0000%                    |
+-----------------------------+
1 row in set (1.14 sec)
```

and for the C++ Sockets Library the output is

Percentage Public Variables for the C++ Sockets Library

```
+-----------------------------+
| Percentage Public Variables |
+-----------------------------+
| 85.6140%                    |
+-----------------------------+
1 row in set (0.18 sec)
```

A-2.8 Set of Queries to Calculate Percentage of Files Referenced

The output for the libodbc++ project for this query is

The Percent Code Used in libodbc++

```
+-------------+
| File_Weight |
+-------------+
|           1 |
|           7 |
|           1 |
|           1 |
|           1 |
|           1 |
|          10 |
|           1 |
|           4 |
|           1 |
|           1 |
|           2 |
|           1 |
|           2 |
|          28 |
|           6 |
|          14 |
|           4 |
|           7 |
|           3 |
|          21 |
|          19 |
|           7 |
|           9 |
|           2 |
|           2 |
|           2 |
|           1 |
|           3 |
|           2 |
+-------------+
30 rows in set (0.19 sec)

+--------------------------------+
| Number Files With Dependencies |
```

```
+---------------------------------+
|                              30 |
+---------------------------------+
1 row in set (0.00 sec)

+---------------------------------+
| Percentage of Files Referenced |
+---------------------------------+
| 71.4286%                        |
+---------------------------------+
1 row in set (0.00 sec)
```

for the log4cxx project the output is

The Percent Code Used in log4cxx

```
+-------------+
| File_Weight |
+-------------+
|           3 |
|           3 |
|           3 |
|           9 |
|          56 |
|           3 |
|           3 |
|           2 |
|          40 |
|           4 |
|           4 |
|           4 |
|           3 |
|           1 |
|           1 |
|           1 |
|           3 |
|           3 |
|           1 |
|           2 |
|           1 |
|         174 |
|           4 |
|         175 |
|           3 |
|         178 |
|           6 |
```

```
|          20 |
|         176 |
|         176 |
|          45 |
|          57 |
|         174 |
|          13 |
|         134 |
|           5 |
|           4 |
|           7 |
|         186 |
|          54 |
|          26 |
|           5 |
|          26 |
|          27 |
|           3 |
|         175 |
|           1 |
|           7 |
|           5 |
|          23 |
|         272 |
|           3 |
|         276 |
|           2 |
|         178 |
|           4 |
|          13 |
|           4 |
|           3 |
|           5 |
|           7 |
|          11 |
|           1 |
|          28 |
|           3 |
|         272 |
|          11 |
|           2 |
|           3 |
|           3 |
|           3 |
|           8 |
```

```
|           4 |
|         193 |
|           4 |
|           6 |
|          19 |
|           6 |
|           4 |
|          20 |
|          16 |
|           9 |
|          20 |
|           4 |
|         262 |
|           4 |
|          41 |
|         270 |
|           4 |
|         306 |
|           9 |
|           6 |
|           3 |
|           3 |
|           5 |
|           2 |
|         176 |
|           5 |
|         183 |
|          17 |
|           3 |
|         100 |
|         178 |
|         221 |
|           3 |
|          29 |
|         317 |
|          19 |
|           5 |
|          21 |
|           4 |
|          10 |
|          29 |
|           4 |
|          21 |
|           9 |
|           4 |
```

```
|           4 |
|           4 |
|           4 |
|           4 |
|           5 |
|           3 |
|          16 |
|           3 |
|           4 |
|           3 |
|          18 |
|           3 |
|          44 |
|           3 |
|           3 |
|           3 |
|           3 |
|           3 |
|           5 |
|           4 |
|           7 |
|           9 |
|           3 |
|           3 |
|           3 |
|          58 |
|           3 |
|           5 |
|           3 |
|           3 |
|           3 |
|          16 |
|         177 |
|          11 |
|           4 |
|           7 |
|           1 |
|           1 |
|          16 |
|           3 |
|           3 |
|           3 |
|           7 |
|           3 |
|           4 |
```

```
|          3 |
|        306 |
|          3 |
|        176 |
|          2 |
|         39 |
|        184 |
|          3 |
|          4 |
|          2 |
|        174 |
|          5 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
|          1 |
```

```
211 |                 1  |
212 |                 1  |
213 |                 1  |
214 |                 1  |
215 |                 1  |
216 |                 1  |
217 |                 1  |
218 |                 1  |
219 |                 1  |
220 |                 1  |
221 |                 1  |
222 |                 1  |
223 |                 1  |
224 |                 1  |
225 |                 1  |
226 |                 1  |
227 |                 1  |
228 |                 1  |
229 |                 1  |
230 |                 1  |
231 |                 1  |
232 |                 1  |
233 |                 1  |
234 |                 1  |
235 |                 1  |
236 |                 1  |
237 |                 1  |
238 |                 1  |
239 |                 1  |
240 |                 1  |
241 |                 1  |
242 |                 1  |
243 |                 1  |
244 |                 1  |
245 |                 1  |
246 |                 1  |
247 |                 1  |
248 |                 1  |
249 |                 1  |
250 |                 1  |
251 |                 1  |
252 |                 1  |
253 |                 1  |
254 |                 1  |
255 |                 1  |
```

```
256 |               1  |
257 |               1  |
258 |               1  |
259 |               1  |
260 |               1  |
261 |               1  |
262 |               1  |
263 |               1  |
264 |               1  |
265 |               1  |
266 |               1  |
267 |               1  |
268 |               1  |
269 |               1  |
270 |               1  |
271 |               1  |
272 |               1  |
273 |               1  |
274 |               1  |
275 |               1  |
276 |               1  |
277 |               1  |
278 |               1  |
279 |               1  |
280 |               1  |
281 |               1  |
282 |               1  |
283 |               1  |
284 |               1  |
285 |               1  |
286 |               1  |
287 |               1  |
288 |               1  |
289 |               1  |
290 |               1  |
291 |               1  |
292 |               1  |
293 |               1  |
294 |               1  |
295 |               1  |
296 |               1  |
297 |               1  |
298 |               1  |
299 |               1  |
300 |               1  |
```

```
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
|            1 |
+--------------+
318 rows in set (0.00 sec)

+---------------------------------+
| Number Files With Dependencies |
+---------------------------------+
|                            318 |
+---------------------------------+
1 row in set (0.00 sec)


+---------------------------------+
|  Percentage of Files Referenced |
+---------------------------------+
|  97.8462%                       |
+--_-----------------------------+
1 row in set (0.00 sec)
```

and for the C++ Sockets Library the output is

The Percent Code Used in the C++ Sockets Library

```
+--------------+
| File_Weight |
+--------------+
```

```
|              4 |
|              5 |
|              6 |
|              6 |
|              2 |
|              2 |
|              2 |
|              2 |
|              2 |
|             13 |
|              4 |
|              9 |
|              7 |
|              2 |
|              2 |
|              4 |
|              2 |
|              2 |
|              2 |
|              4 |
|              3 |
|              4 |
|              2 |
|              6 |
|              4 |
|              2 |
|             50 |
|             10 |
|              2 |
|              2 |
|              4 |
|              2 |
|              2 |
|              2 |
|             40 |
|              2 |
|              9 |
|              2 |
|             16 |
|             23 |
|              4 |
|              6 |
|             52 |
|              2 |
|              1 |
```

```
|           30 |
|            2 |
|            2 |
|            2 |
|           32 |
|            2 |
|           75 |
|           49 |
|           29 |
|            2 |
|            2 |
|           26 |
|           10 |
|            4 |
|            6 |
|            2 |
|            2 |
|            8 |
|            2 |
|            2 |
|            2 |
|            2 |
|           10 |
|            2 |
|            5 |
|           46 |
|            2 |
|           43 |
|            6 |
|            2 |
|            2 |
|           35 |
|            2 |
|            2 |
|           16 |
|           47 |
|            2 |
|            2 |
+--------------+
83 rows in set (1.65 sec)

+--------------------------------+
| Number Files With Dependencies |
+--------------------------------+
|                             83 |
```

```
+---------------------------------+
1 row in set (0.00 sec)

+---------------------------------+
| Percentage of Files Referenced |
+---------------------------------+
| 68.0327868852459%              |
+---------------------------------+
1 row in set (0.00 sec)
```

A-2.9 Percent Code Reused

The output for the libodbc++ project for this query is

Percent Code Reused by libodbc++

```
+----------------------+
| Percent Code Reused |
+----------------------+
| 404.7619%           |
+----------------------+
1 row in set (0.93 sec)
```

for the log4cxx project the output is

Percent Code Reused by log4cxx

```
+----------------------+
| Percent Code Reused |
+----------------------+
| 48.9231%            |
+----------------------+
1 row in set (1 min 3.72 sec)
```

and for the C++ Sockets Library the output is

Percent Code Reused by the C++ Sockets Library

```
+----------------------+
| Percent Code Reused |
+----------------------+
| 1.1393%             |
+----------------------+
1 row in set (8.56 sec)
```

A-3 Advanced Query Output

A-3.1 Search Source for a String

Search Source for 'Event::Data() const'

name	id	file_id	located_at_char
Event.h	1	1	0
Debug.cpp	2	2	0
IHttpServer.h	3	3	0
SocketStream.cpp	4	4	0
HttpRequest.h	5	5	0
EventTime.h	6	6	0
HttpRequest.cpp	7	7	0
HttpPostSocket.h	8	8	0
Thread.cpp	9	9	0
HttpBaseSocket.h	10	10	0
MemFile.cpp	11	11	0
Ajp13Socket.cpp	12	12	0
HttpPostSocket.cpp	13	13	0
XmlException.cpp	14	14	0
SocketThread.cpp	15	15	0
EventHandler.h	16	16	0
Exception.h	17	17	0
IEventOwner.h	18	18	0
Lock.h	19	19	0
Sockets-config.cpp	20	20	0
Ipv4Address.h	21	21	0
ResolvServer.cpp	22	22	0
StreamSocket.cpp	23	23	0
SctpSocket.h	24	24	0
HttpResponse.cpp	25	25	0
EventHandler.cpp	26	26	0
SocketStream.h	27	27	0
SocketHandlerEp.cpp	28	28	0
UdpSocket.cpp	29	29	0
Mutex.cpp	30	30	0
UdpSocket.h	31	31	0
StdoutLog.cpp	32	32	0
ResolvSocket.cpp	33	33	0
StreamWriter.h	34	34	0
HttpdCookies.cpp	35	35	0
FileStream.h	36	36	0

File.cpp	37	37	0
Utility.cpp	38	38	0
XmlDocument.h	39	39	0
sockets-config.h	40	40	0
EventTime.cpp	41	41	0
HTTPSocket.cpp	42	42	0
SctpSocket.cpp	43	43	0
SocketThread.h	44	44	0
socket_include.cpp	45	45	0
Parse.cpp	46	46	0
IFileUpload.h	47	47	0
AjpBaseSocket.h	48	48	0
Debug.h	49	49	0
HttpdForm.h	50	50	0
SocketHandlerThread.h	51	51	0
SSLInitializer.cpp	52	52	0
Ipv6Address.h	53	53	0
SocketHandlerThread.cpp	54	54	0
SmtpdSocket.cpp	55	55	0
XmlNode.h	56	56	0
Thread.h	57	57	0
Base64.cpp	58	58	0
SocketHandler.h	59	59	0
Lock.cpp	60	60	0
XmlException.h	61	61	0
HttpTransaction.cpp	62	62	0
FileStream.cpp	63	63	0
Base64.h	64	64	0
TcpSocket.cpp	65	65	0
StreamWriter.cpp	66	66	0
SocketHandler.cpp	67	67	0
Json.h	68	68	0
Mutex.h	69	69	0
Exception.cpp	70	70	0
IStream.h	71	71	0
HttpBaseSocket.cpp	72	72	0
HTTPSocket.h	73	73	0
IFile.h	74	74	0
ListenSocket.h	75	75	0
ResolvSocket.h	76	76	0
HttpResponse.h	77	77	0
Semaphore.h	78	78	0
IEventOwner.cpp	79	79	0
IBase.h	80	80	0
TcpSocket.h	81	81	0

```
| Semaphore.cpp           |  82 |      82 |               0 |
| XmlDocument.cpp         |  83 |      83 |               0 |
| HttpDebugSocket.h       |  84 |      84 |               0 |
| AjpBaseSocket.cpp       |  85 |      85 |               0 |
| SSLInitializer.h        |  86 |      86 |               0 |
| HttpDebugSocket.cpp     |  87 |      87 |               0 |
| socket_include.h        |  88 |      88 |               0 |
| SocketAddress.h         |  89 |      89 |               0 |
| StdLog.h                |  90 |      90 |               0 |
| HttpdSocket.h           |  91 |      91 |               0 |
| HttpPutSocket.h         |  92 |      92 |               0 |
| ISocketHandler.h        |  93 |      93 |               0 |
| HttpdCookies.h          |  94 |      94 |               0 |
| ResolvServer.h          |  95 |      95 |               0 |
| Ipv6Address.cpp         |  96 |      96 |               0 |
| XmlNode.cpp             |  97 |      97 |               0 |
| MemFile.h               |  98 |      98 |               0 |
| SmtpdSocket.h           |  99 |      99 |               0 |
| HttpGetSocket.h         | 100 |     100 |               0 |
| HttpClientSocket.h      | 101 |     101 |               0 |
| Ajp13Socket.h           | 102 |     102 |               0 |
| Event.cpp               | 103 |     103 |            1860 |
| HttpPutSocket.cpp       | 104 |     104 |               0 |
| HttpdSocket.cpp         | 105 |     105 |               0 |
| HttpTransaction.h       | 106 |     106 |               0 |
| Ipv4Address.cpp         | 107 |     107 |               0 |
| IEventHandler.h         | 108 |     108 |               0 |
| Socket.h                | 109 |     109 |               0 |
| HttpClientSocket.cpp    | 110 |     110 |               0 |
| IMutex.h                | 111 |     111 |               0 |
| Parse.h                 | 112 |     112 |               0 |
| Json.cpp                | 113 |     113 |               0 |
| Socket.cpp              | 114 |     114 |               0 |
| StreamSocket.h          | 115 |     115 |               0 |
| StdoutLog.h             | 116 |     116 |               0 |
| HttpdForm.cpp           | 117 |     117 |               0 |
| File.h                  | 118 |     118 |               0 |
| ajp13.h                 | 119 |     119 |               0 |
| Utility.h               | 120 |     120 |               0 |
| SocketHandlerEp.h       | 121 |     121 |               0 |
| HttpGetSocket.cpp       | 122 |     122 |               0 |
+-------------------------+-----+---------+-----------------+
122 rows in set (0.06 sec)
```

A-3.2 Search XML for a String

Search XML for 'Event::Data() const'

name	id	file_id	located_at_char
Event.xml	1	123	63660
Debug.xml	2	124	0
IHttpServer.xml	3	125	0
SocketStream.xml	4	126	0
HttpRequest.xml	5	127	0
EventTime.xml	6	128	0
HttpPostSocket.xml	7	129	0
Thread.xml	8	130	0
HttpBaseSocket.xml	9	131	0
MemFile.xml	10	132	0
Ajp13Socket.xml	11	133	0
XmlException.xml	12	134	0
SocketThread.xml	13	135	0
EventHandler.xml	14	136	0
Exception.xml	15	137	0
IEventOwner.xml	16	138	0
Lock.xml	17	139	0
Ipv4Address.xml	18	141	0
ResolvServer.xml	19	142	0
StreamSocket.xml	20	143	0
SctpSocket.xml	21	144	0
HttpResponse.xml	22	145	0
SocketHandlerEp.xml	23	146	0
UdpSocket.xml	24	147	0
Mutex.xml	25	148	0
StdoutLog.xml	26	149	0
ResolvSocket.xml	27	150	0
StreamWriter.xml	28	151	0
HttpdCookies.xml	29	152	0
FileStream.xml	30	153	0
File.xml	31	154	0
Utility.xml	32	155	0
XmlDocument.xml	33	156	0
HTTPSocket.xml	34	157	0
socket_include.xml	35	158	0
Parse.xml	36	159	0
IFileUpload.xml	37	160	0
AjpBaseSocket.xml	38	161	0
HttpdForm.xml	39	162	0

```
| SocketHandlerThread.xml | 40 |     163 |               0 |
| SSLInitializer.xml      | 41 |     164 |               0 |
| Ipv6Address.xml         | 42 |     165 |               0 |
| SmtpdSocket.xml         | 43 |     166 |               0 |
| XmlNode.xml             | 44 |     167 |               0 |
| Base64.xml              | 45 |     168 |               0 |
| SocketHandler.xml       | 46 |     169 |               0 |
| HttpTransaction.xml     | 47 |     170 |               0 |
| TcpSocket.xml           | 48 |     171 |               0 |
| Json.xml                | 49 |     172 |               0 |
| IStream.xml             | 50 |     173 |               0 |
| IFile.xml               | 51 |     174 |               0 |
| ListenSocket.xml        | 52 |     175 |               0 |
| Semaphore.xml           | 53 |     176 |               0 |
| IBase.xml               | 54 |     177 |               0 |
| HttpDebugSocket.xml     | 55 |     178 |               0 |
| SocketAddress.xml       | 56 |     179 |               0 |
| StdLog.xml              | 57 |     180 |               0 |
| HttpdSocket.xml         | 58 |     181 |               0 |
| HttpPutSocket.xml       | 59 |     182 |               0 |
| ISocketHandler.xml      | 60 |     183 |               0 |
| HttpGetSocket.xml       | 61 |     184 |               0 |
| HttpClientSocket.xml    | 62 |     185 |               0 |
| IEventHandler.xml       | 63 |     186 |               0 |
| Socket.xml              | 64 |     187 |               0 |
| IMutex.xml              | 65 |     188 |               0 |
| ajp13.xml               | 66 |     189 |               0 |
+-------------------------+----+---------+-----------------+
66 rows in set (8.36 sec)
```

A-3.3 Find Code Surrounding a String in Source

Find Code Surrounding 'Event::Data() const' in Source

```
+---------+----+---------+----------------------+
| name    | id | file_id | data                 |
+---------+----+---------+----------------------+
| Event.h |  1 |       1 | / _SOCKETS_Event_H


 |
+---------+----+---------+----------------------+
1 row in set (0.00 sec)
```

A-3.4 Find Code Surrounding a String in XML

Find Code Surrounding 'Event::Data() const' in XML

```
+-----------+----+---------+-----------------------------------------+
| name      | id | file_id | data                                    |
+-----------+----+---------+-----------------------------------------+
| Event.xml |  1 |     123 | 4DataEv" demangled="Event::Data() const" |
+-----------+----+---------+-----------------------------------------+
1 row in set (0.01 sec)
```