Dependency Extraction

Team Space Jam

Overview

- Dependency Extraction using Understand
- Dependency Extraction using include directives
- Dependency Extraction using srcML
- Qualitative comparison between the extraction techniques
- Quantitative comparison between the extraction techniques
- Use case and sequence diagrams
- Limitations of reported findings
- Lessons Learned

Dependency Extraction using Understand

What is Understand ?

- A static analysis tool used with a focus on standards testing, metrics and source code comprehension.
- It is essentially used to maintain and understand large amounts of source code, be it legacy or newly created.
- Provides the user with a multi-language, cross-platform, maintenance-oriented Interactive Development Environment (IDE)
- Its services can be used on source code in the following languages: C, C++, C#, Objective C/Objective C++, Ada, Java, Pascal/Delphi, COBOL, JOVIAL, VHDL, Fortran, PL/M, Python, PHP, HTML, CSS, JavaScript, and XML.

Features

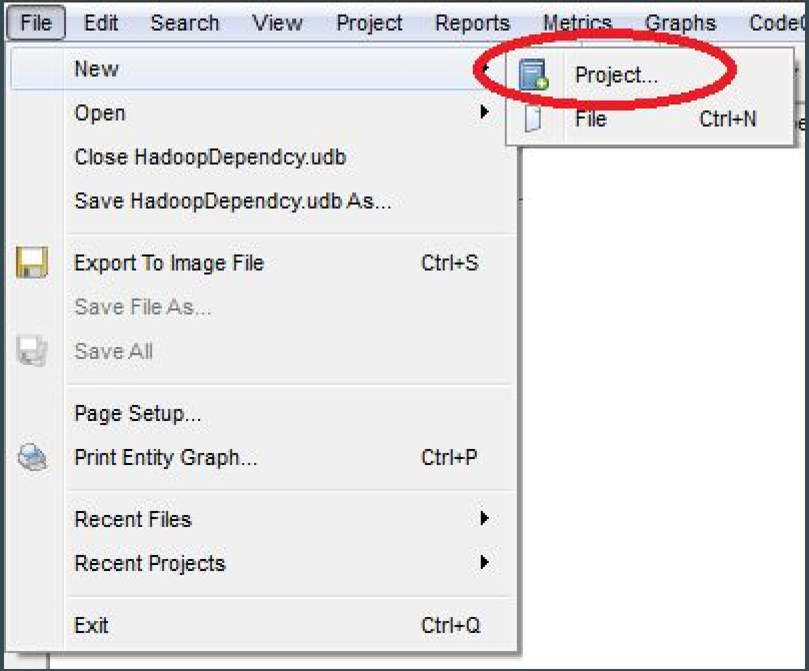
- Code Knowledge
- Metrics and Reports
- Graphing
- Standards Testing
- Dependency Analysis
- Editor
- Searches
- Languages

Dependency Analysis

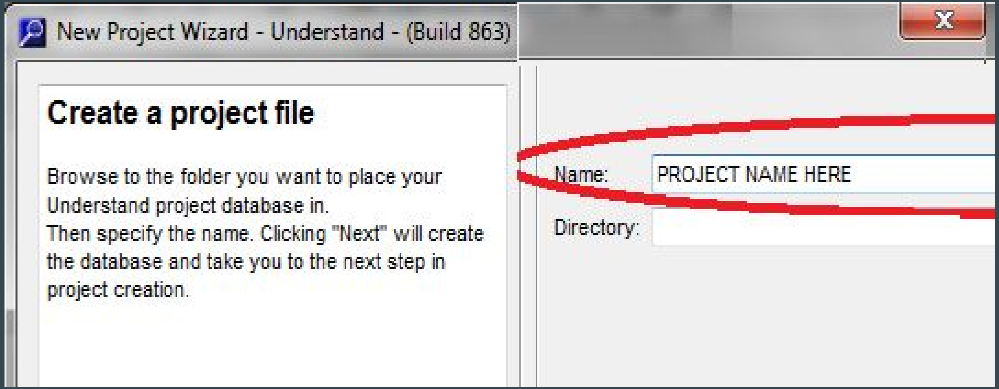
- How ?
  - Every reference in the understand project is examined. Then a dependency data structure is built for every file and architecture. This structure will include the references that caused the dependency and the nature of said dependency. Due to the amount of data being large, this is not calculated as you ask for a certain dependency relationship. Rather, all dependency information is calculated and cached, which results in quick exploration and browsing.
- Capabilities
  - Quick navigation of dependencies for project architectures and files
  - Graphs, Depends On and Dependents for architectures and files
  - Dependency relationships exportation to spreasheet
  - Dependency Browsing tool used to show all dependencys and their information

Using Understand

Create New Project



Name it


New Project Wizard - Understand - (Build 863)

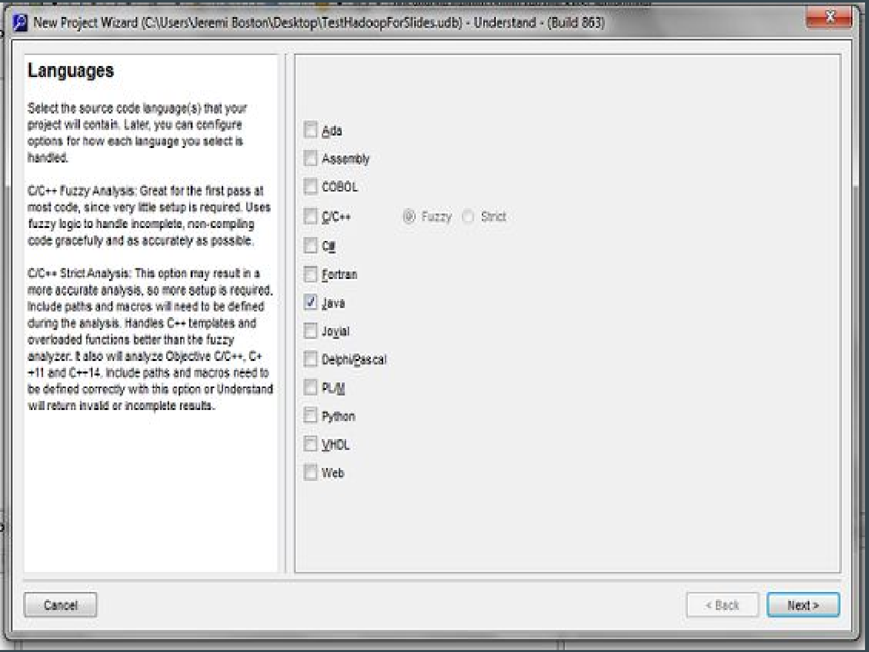
Create a project file

Browse to the folder you want to place your Understand project database in.
Then specify the name. Clicking "Next" will create the database and take you to the next step in project creation.

Name: PROJECT NAME HERE

Directory:

Select source code language



New Project Wizard (C:\Users\Jeremi Boston\Desktop\TestHadoopForSlides.udb) - Understand - (Build 863)

Languages

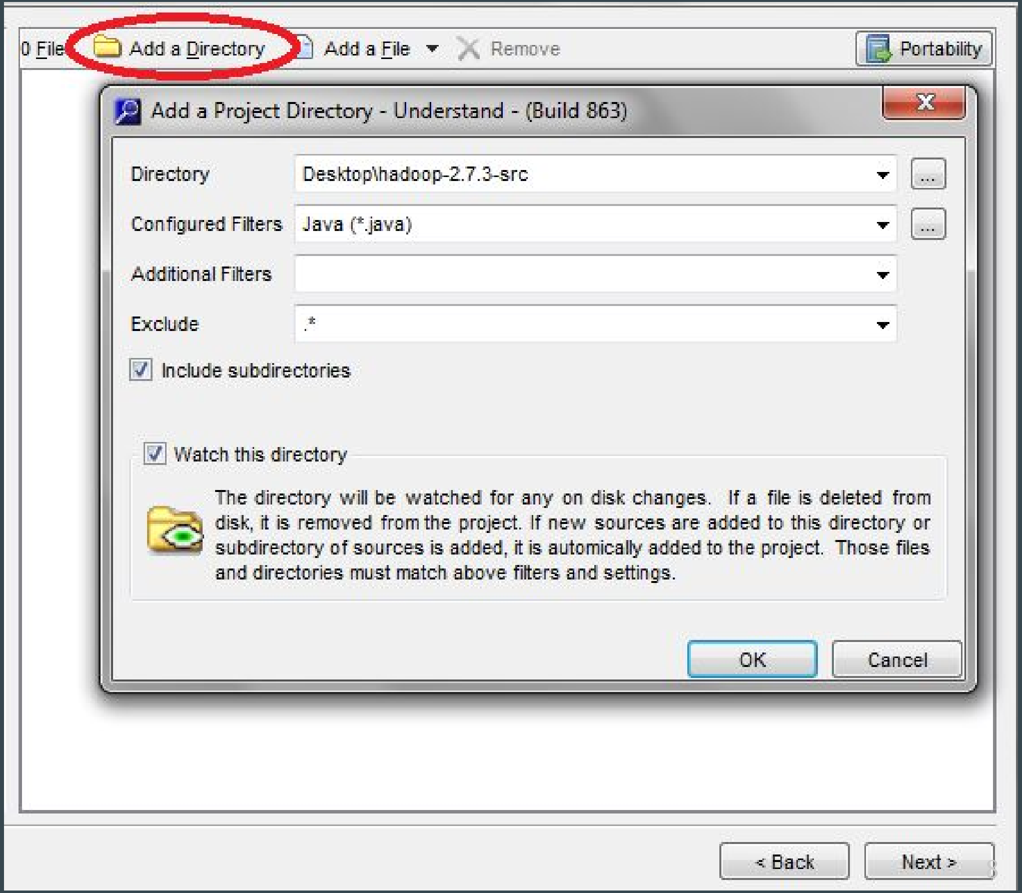
Select the source code language(s) that your project will contain. Later, you can configure options for how each language you select is handled.

C/C++ Fuzzy Analysis: Great for the first pass at most code, since very little setup is required. Uses fuzzy logic to handle incomplete, non-compiling code gracefully and as accurately as possible.

C/C++ Strict Analysis: This option may result in a more accurate analysis, so more setup is required. Include paths and macros will need to be defined during the analysis. Handles C++ templates and overloaded functions better than the fuzzy analyzer. It also will analyze Objective C/C++, C++11 and C++14. Include paths and macros need to be defined correctly with this option or Understand will return invalid or incomplete results.

- ☐ Ada
- ☐ Assembly
- ☐ COBOL
- ☐ C/C++ [✓ Fuzzy] Strict
- ☐ C#
- ☐ Fortran
- ☑ Java
- ☐ Jovial
- ☐ Delphi/Pascal
- ☐ PL/M
- ☐ Python
- ☐ VHDL
- ☐ Web

Add source code directory



Add a Project Directory - Understand - (Build 863)

Directory: Desktop\hadoop-2.7.3-src

Configured Filters: Java (*.java)

Additional Filters:

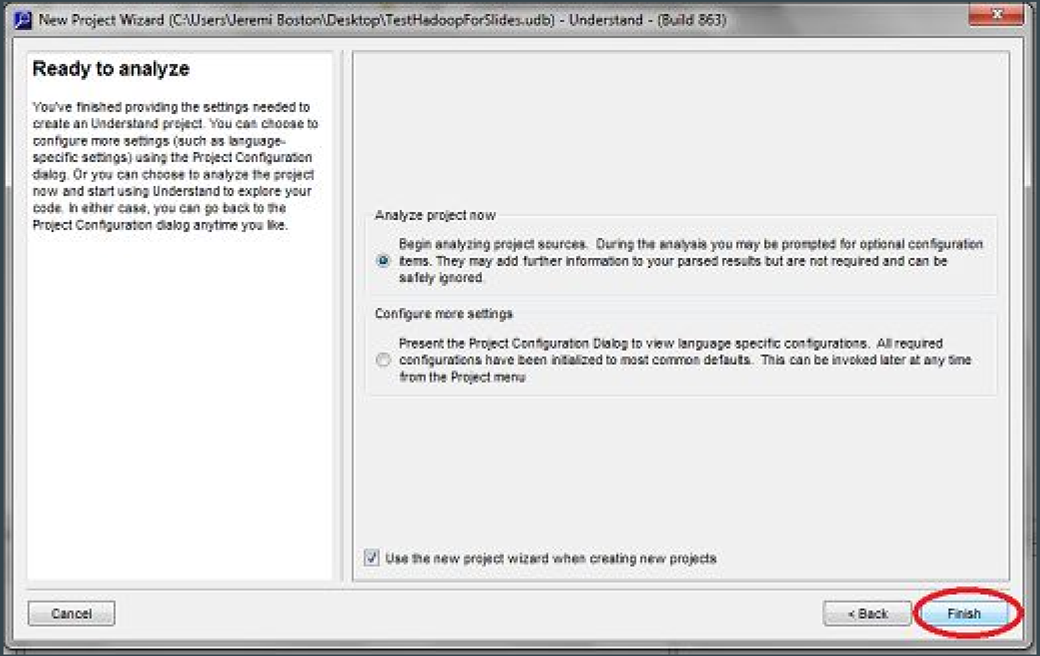
Exclude: .*

☑ Include subdirectories

☑ Watch this directory

The directory will be watched for any on disk changes. If a file is deleted from disk, it is removed from the project. If new sources are added to this directory or subdirectory of sources is added, it is automically added to the project. Those files and directories must match above filters and settings.

Understand Project Analysis



New Project Wizard (C:\Users\Jeremi Boston\Desktop\TestHadoopForSlides.udb) - Understand - (Build 863)

Ready to analyze

You've finished providing the settings needed to create an Understand project. You can choose to configure more settings (such as language-specific settings) using the Project Configuration dialog. Or you can choose to analyze the project now and start using Understand to explore your code. In either case, you can go back to the Project Configuration dialog anytime you like.

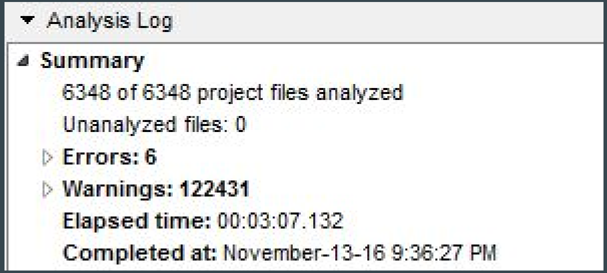
Analyze project now

● Begin analyzing project sources. During the analysis you may be prompted for optional configuration items. They may add further information to your parsed results but are not required and can be safely ignored.

Configure more settings

○ Present the Project Configuration Dialog to view language specific configurations. All required configurations have been initialized to most common defaults. This can be invoked later at any time from the Project menu

☑ Use the new project wizard when creating new projects



Analysis Log

Summary

6348 of 6348 project files analyzed

Unanalyzed files: 0

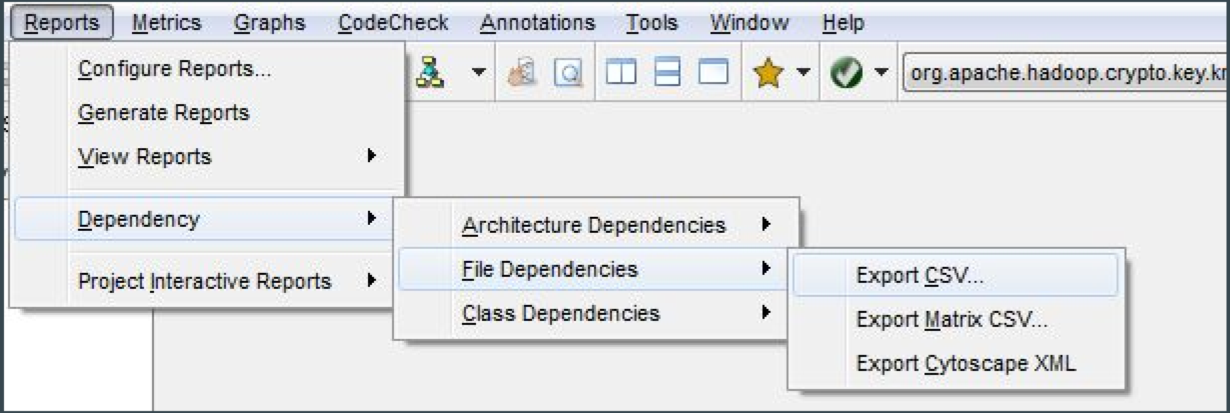
Errors: 6

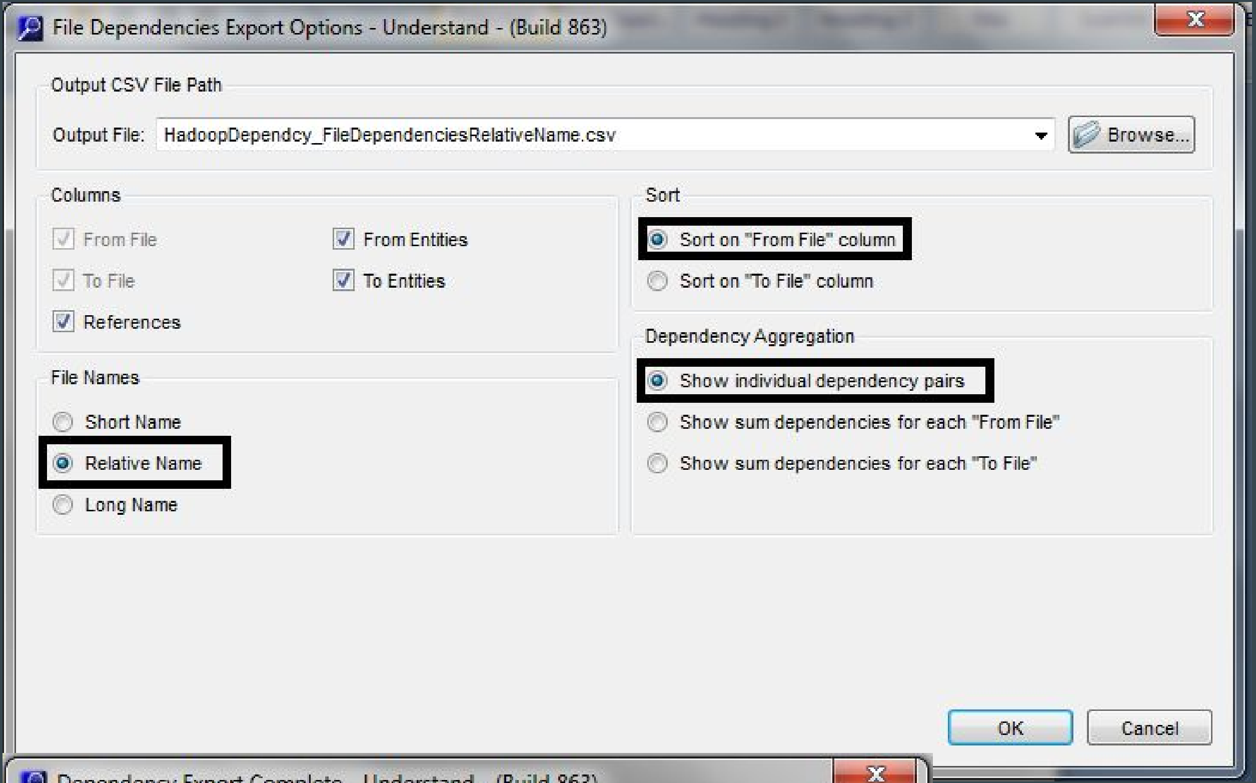
Warnings: 122431

Elapsed time: 00:03:07.132

Completed at: November-13-16 9:36:27 PM

Export the dependencies report to CSV





File Dependencies Export Options - Understand - (Build 863)

Output CSV File Path

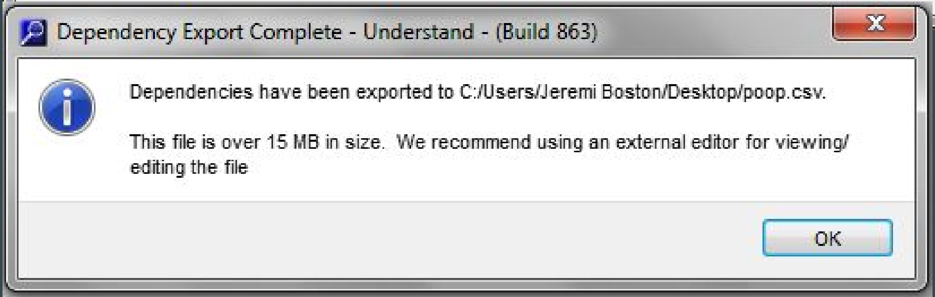
Output File: HadoopDependcy_FileDependenciesRelativeName.csv

Columns
- [✓] From File
- [✓] To File
- [✓] References
- [✓] From Entities
- [✓] To Entities

Sort
- (●) Sort on "From File" column
- ( ) Sort on "To File" column

File Names
- ( ) Short Name
- (●) Relative Name
- ( ) Long Name

Dependency Aggregation
- (●) Show individual dependency pairs
- ( ) Show sum dependencies for each "From File"
- ( ) Show sum dependencies for each "To File"



Dependency Export Complete - Understand - (Build 863)

Dependencies have been exported to C:/Users/Jeremi Boston/Desktop/poop.csv.

This file is over 15 MB in size. We recommend using an external editor for viewing/editing the file

Select Dependencies Export Options

We are done !

Dependency Extraction using Include Directives

Dependency Extraction using Include Directives

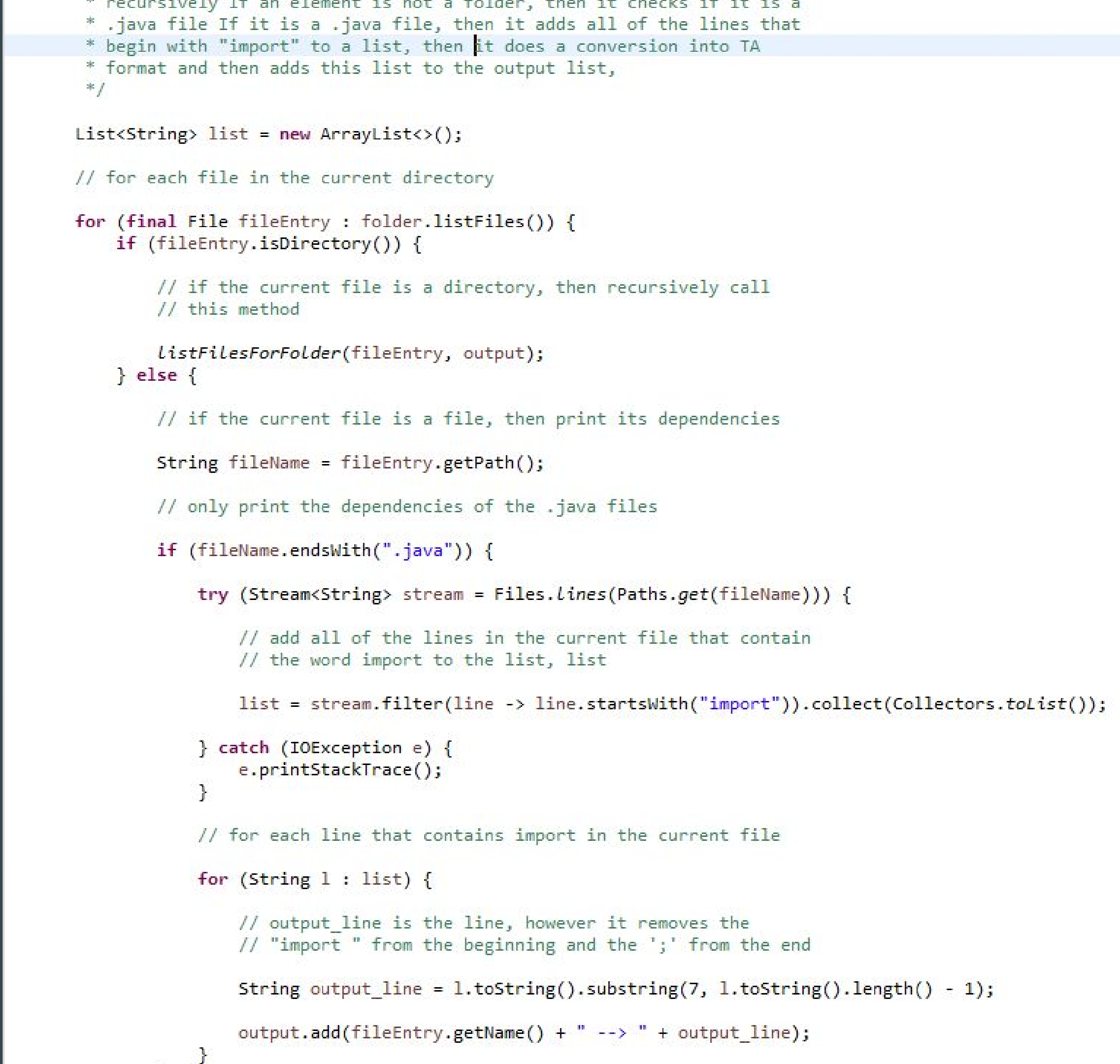
- General concept
  - Go through all of the java files in the hadoop directory
  - Find all instances of the “import” statement
  - For each import statement
    - Add C -> D
      - Where C is the current file, and D is what the file depends on

Dependency Extraction using Include Directives

Pseudocode

- For each .java file in the hadoop directory
- Find all of the lines that use import
- Add these lines in the form of C -> D

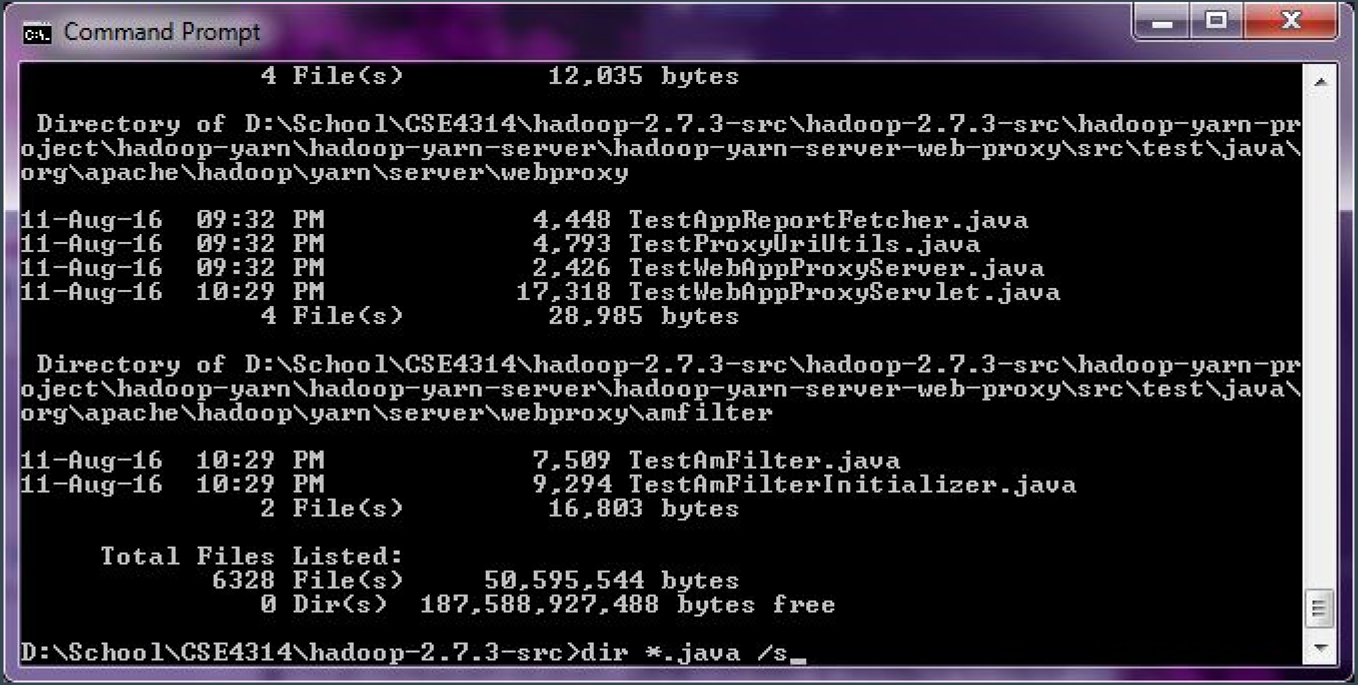
Dependency Extraction using Include Directives



```
 * recursively if an element is not a folder, then it checks if it is a
 * .java file If it is a .java file, then it adds all of the lines that
 * begin with "import" to a list, then it does a conversion into TA
 * format and then adds this list to the output list,
 */

List<String> list = new ArrayList<>();

// for each file in the current directory

for (final File fileEntry : folder.listFiles()) {
    if (fileEntry.isDirectory()) {

        // if the current file is a directory, then recursively call
        // this method

        listFilesForFolder(fileEntry, output);
    } else {

        // if the current file is a file, then print its dependencies

        String fileName = fileEntry.getPath();

        // only print the dependencies of the .java files

        if (fileName.endsWith(".java")) {

            try (Stream<String> stream = Files.lines(Paths.get(fileName))) {

                // add all of the lines in the current file that contain
                // the word import to the list, list

                list = stream.filter(line -> line.startsWith("import")).collect(Collectors.toList());

            } catch (IOException e) {
                e.printStackTrace();
            }

            // for each line that contains import in the current file

            for (String l : list) {

                // output_line is the line, however it removes the
                // "import " from the beginning and the ';' from the end

                String output_line = l.toString().substring(7, l.toString().length() - 1);

                output.add(fileEntry.getName() + " --> " + output_line);
            }
```

Dependency Extraction using srcML

- srcML takes a code file as input
- Need to recurse through each subdirectory to feed required files as input to srcML

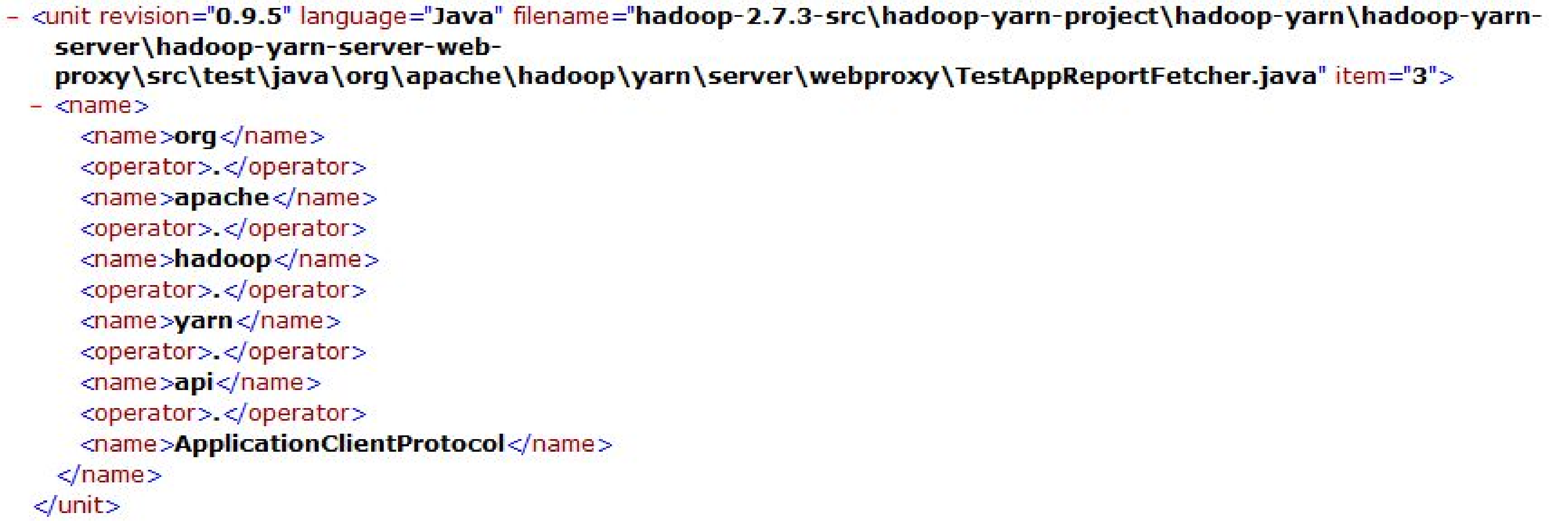


```
               4 File(s)         12,035 bytes

 Directory of D:\School\CSE4314\hadoop-2.7.3-src\hadoop-2.7.3-src\hadoop-yarn-pr
oject\hadoop-yarn\hadoop-yarn-server\hadoop-yarn-server-web-proxy\src\test\java\
org\apache\hadoop\yarn\server\webproxy

11-Aug-16  09:32 PM             4,448 TestAppReportFetcher.java
11-Aug-16  09:32 PM             4,793 TestProxyUriUtils.java
11-Aug-16  09:32 PM             2,426 TestWebAppProxyServer.java
11-Aug-16  10:29 PM            17,318 TestWebAppProxyServlet.java
               4 File(s)         28,985 bytes

 Directory of D:\School\CSE4314\hadoop-2.7.3-src\hadoop-2.7.3-src\hadoop-yarn-pr
oject\hadoop-yarn\hadoop-yarn-server\hadoop-yarn-server-web-proxy\src\test\java\
org\apache\hadoop\yarn\server\webproxy\amfilter

11-Aug-16  10:29 PM             7,509 TestAmFilter.java
11-Aug-16  10:29 PM             9,294 TestAmFilterInitializer.java
               2 File(s)         16,803 bytes

     Total Files Listed:
            6328 File(s)     50,595,544 bytes
               0 Dir(s)  187,588,927,488 bytes free

D:\School\CSE4314\hadoop-2.7.3-src>dir *.java /s_
```

Dependency Extraction using srcML

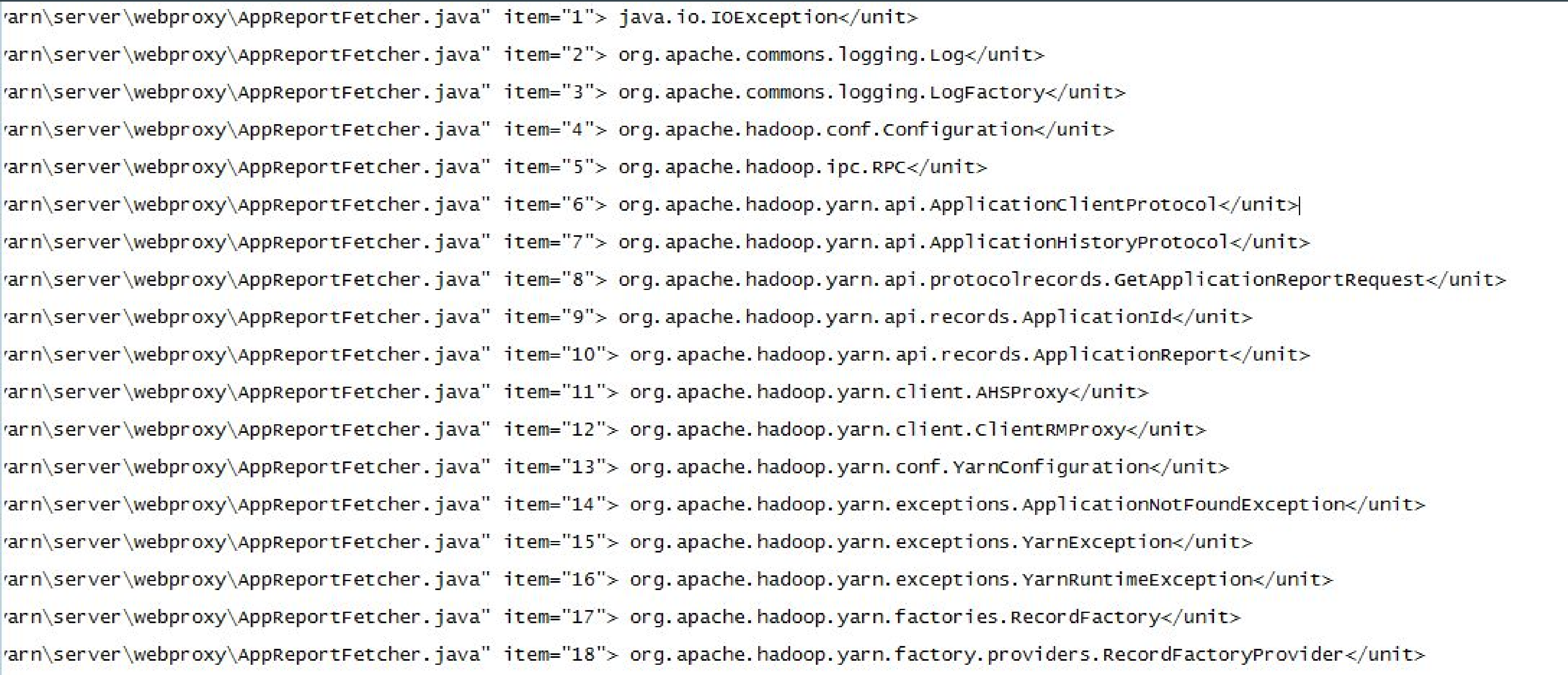
- Output of srcML is a .xml file
- Lists the input file
- Provides # of outward calls and where

- The .xml document can be further parsed to extract only the calls



```
- <unit revision="0.9.5" language="Java" filename="hadoop-2.7.3-src\hadoop-yarn-project\hadoop-yarn\hadoop-yarn-
    server\hadoop-yarn-server-web-
    proxy\src\test\java\org\apache\hadoop\yarn\server\webproxy\TestAppReportFetcher.java" item="3">
  - <name>
      <name>org</name>
      <operator>.</operator>
      <name>apache</name>
      <operator>.</operator>
      <name>hadoop</name>
      <operator>.</operator>
      <name>yarn</name>
      <operator>.</operator>
      <name>api</name>
      <operator>.</operator>
      <name>ApplicationClientProtocol</name>
    </name>
  </unit>
```

Dependency Extraction using srcML



```
arn\server\webproxy\AppReportFetcher.java" item="1"> java.io.IOException</unit>
arn\server\webproxy\AppReportFetcher.java" item="2"> org.apache.commons.logging.Log</unit>
arn\server\webproxy\AppReportFetcher.java" item="3"> org.apache.commons.logging.LogFactory</unit>
arn\server\webproxy\AppReportFetcher.java" item="4"> org.apache.hadoop.conf.Configuration</unit>
arn\server\webproxy\AppReportFetcher.java" item="5"> org.apache.hadoop.ipc.RPC</unit>
arn\server\webproxy\AppReportFetcher.java" item="6"> org.apache.hadoop.yarn.api.ApplicationClientProtocol</unit>
arn\server\webproxy\AppReportFetcher.java" item="7"> org.apache.hadoop.yarn.api.ApplicationHistoryProtocol</unit>
arn\server\webproxy\AppReportFetcher.java" item="8"> org.apache.hadoop.yarn.api.protocolrecords.GetApplicationReportRequest</unit>
arn\server\webproxy\AppReportFetcher.java" item="9"> org.apache.hadoop.yarn.api.records.ApplicationId</unit>
arn\server\webproxy\AppReportFetcher.java" item="10"> org.apache.hadoop.yarn.api.records.ApplicationReport</unit>
arn\server\webproxy\AppReportFetcher.java" item="11"> org.apache.hadoop.yarn.client.AHSProxy</unit>
arn\server\webproxy\AppReportFetcher.java" item="12"> org.apache.hadoop.yarn.client.ClientRMProxy</unit>
arn\server\webproxy\AppReportFetcher.java" item="13"> org.apache.hadoop.yarn.conf.YarnConfiguration</unit>
arn\server\webproxy\AppReportFetcher.java" item="14"> org.apache.hadoop.yarn.exceptions.ApplicationNotFoundException</unit>
arn\server\webproxy\AppReportFetcher.java" item="15"> org.apache.hadoop.yarn.exceptions.YarnException</unit>
arn\server\webproxy\AppReportFetcher.java" item="16"> org.apache.hadoop.yarn.exceptions.YarnRuntimeException</unit>
arn\server\webproxy\AppReportFetcher.java" item="17"> org.apache.hadoop.yarn.factories.RecordFactory</unit>
arn\server\webproxy\AppReportFetcher.java" item="18"> org.apache.hadoop.yarn.factory.providers.RecordFactoryProvider</unit>
```

Dependency Extraction using srcML

Advantages

- Can parse through an input file to search for **ANY** keywords
- Lists # of specified item found
  - Can measure any metric you want!
- Output .xml file can be further parsed to enhance your findings
- XML documents are machine-readable and supply human-readable tags

Disadvantages

- Unable to determine calls from file X.java **TO** your input file
- XML files can be quickly cluttered with <tags>
- Only supported languages are C, C#, C++, Java

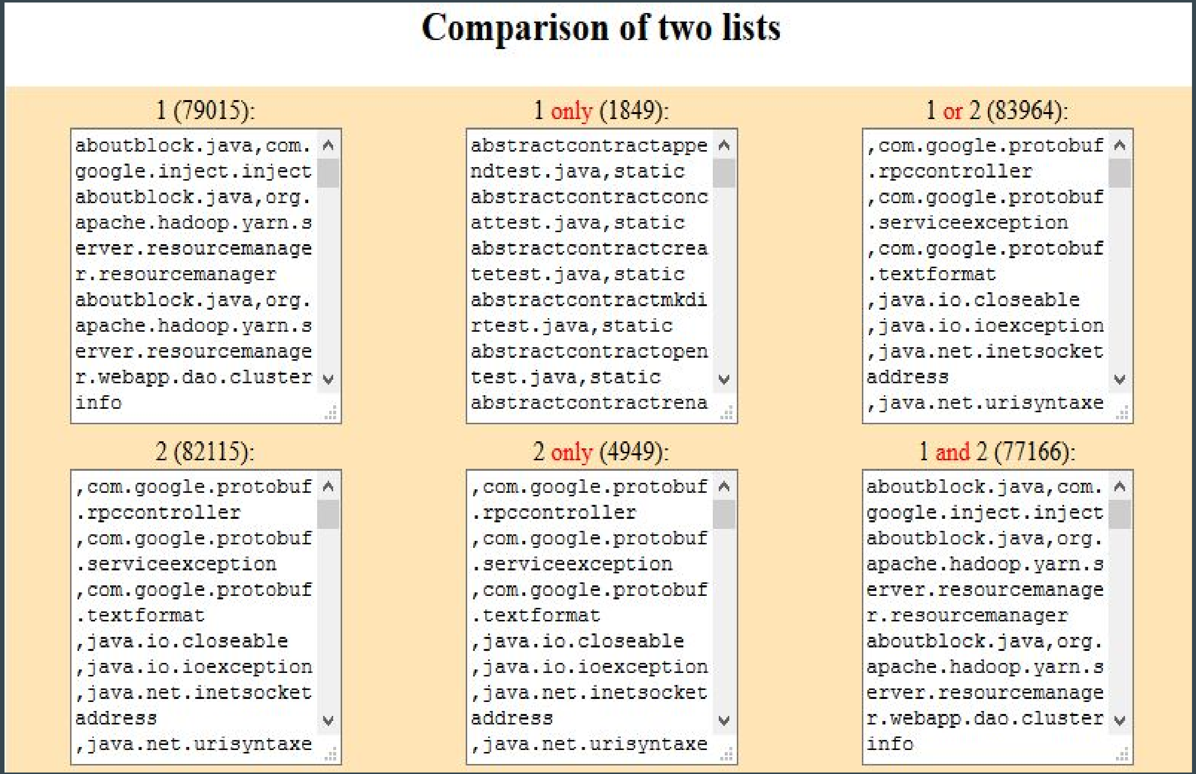
Comparison - Quantitative Results

- Three tools were used to find the dependencies in hadoop
- Understand found around 83,000 class dependencies and around 62,000 file dependencies
- Include discovered close to 84,000 file dependencies
- srcML found about 87,000 file dependencies

Comparison - Quantitative Results

Include(1) vs. srcML(2)

- Unique to include 1849 entries
- Unique to srcML 4949 entries
- Overlap between Include and srcML contains 77166 entries



Comparison of two lists

1 (79015):

```
aboutblock.java,com.
google.inject.inject
aboutblock.java,org.
apache.hadoop.yarn.s
erver.resourcemanage
r.resourcemanager
aboutblock.java,org.
apache.hadoop.yarn.s
erver.resourcemanage
r.webapp.dao.cluster
info
```

1 only (1849):

```
abstractcontractappe
ndtest.java,static
abstractcontractconc
attest.java,static
abstractcontractcrea
tetest.java,static
abstractcontractmkdi
rtest.java,static
abstractcontractopen
test.java,static
abstractcontractrena
```

1 or 2 (83964):

```
,com.google.protobuf
.rpccontroller
,com.google.protobuf
.serviceexception
,com.google.protobuf
.textformat
,java.io.closeable
,java.io.ioexception
,java.net.inetsocket
address
,java.net.urisyntaxe
```

2 (82115):

```
,com.google.protobuf
.rpccontroller
,com.google.protobuf
.serviceexception
,com.google.protobuf
.textformat
,java.io.closeable
,java.io.ioexception
,java.net.inetsocket
address
,java.net.urisyntaxe
```

2 only (4949):

```
,com.google.protobuf
.rpccontroller
,com.google.protobuf
.serviceexception
,com.google.protobuf
.textformat
,java.io.closeable
,java.io.ioexception
,java.net.inetsocket
address
,java.net.urisyntaxe
```

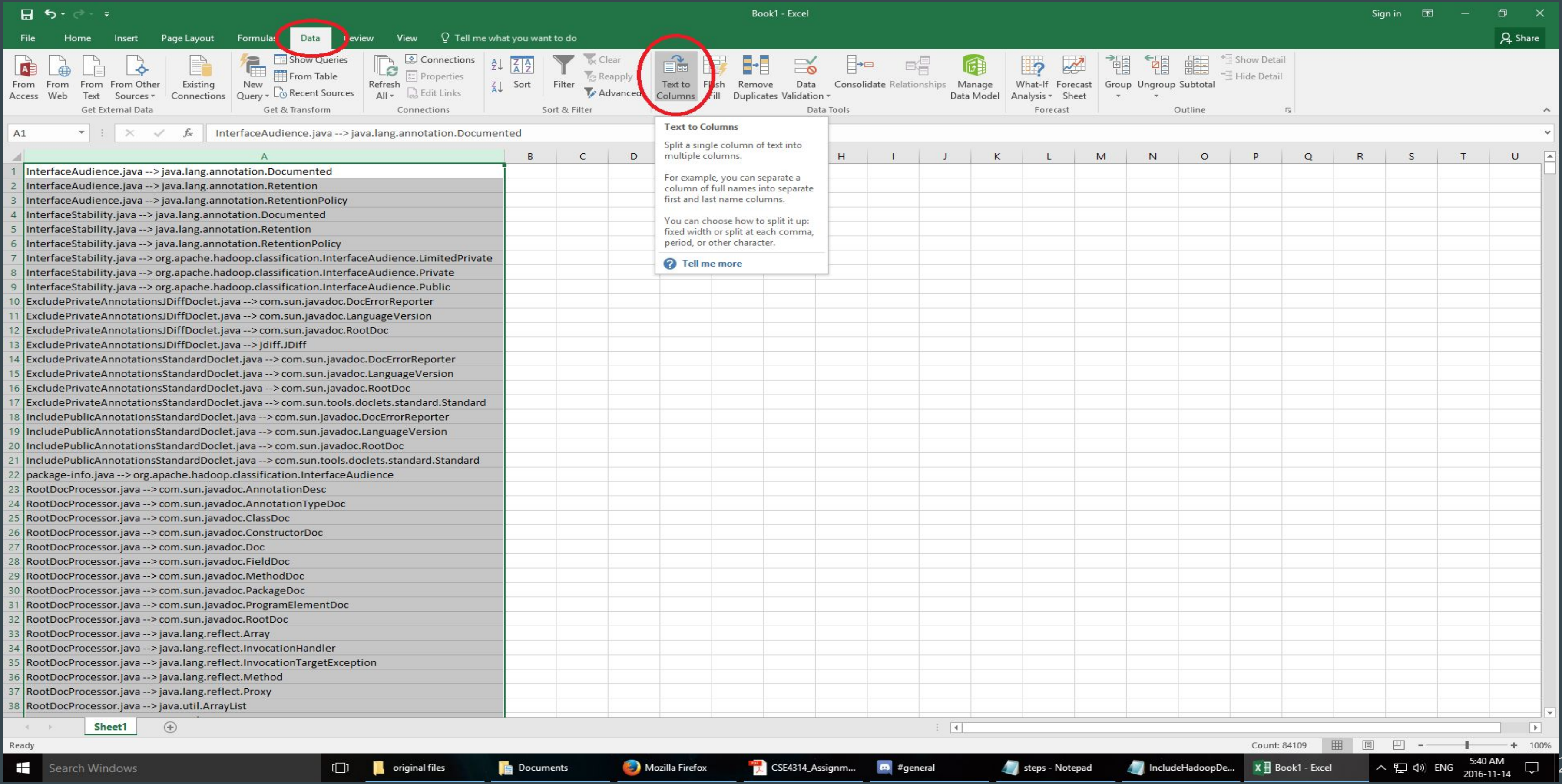
1 and 2 (77166):

```
aboutblock.java,com.
google.inject.inject
aboutblock.java,org.
apache.hadoop.yarn.s
erver.resourcemanage
r.resourcemanager
aboutblock.java,org.
apache.hadoop.yarn.s
erver.resourcemanage
r.webapp.dao.cluster
info
```

Comparison - Process

- Gather data from all dependency tools used
- Use excel to format data
- Once all data contains the exact same format comparison begins
- Take all entries and use software to compare the two lists of data from separate tools
- Once completed use sample size calculator
- Confidence level of: 95%, Confidence interval: 5, Population: 78,000, indicating a required sample size of 382 random dependencies
- Use random number generator in excel to shuffle entries
- Take the first 382 samples

Comparison - Process



InterfaceAudience.java --> java.lang.annotation.Documented

Text to Columns

Split a single column of text into multiple columns.

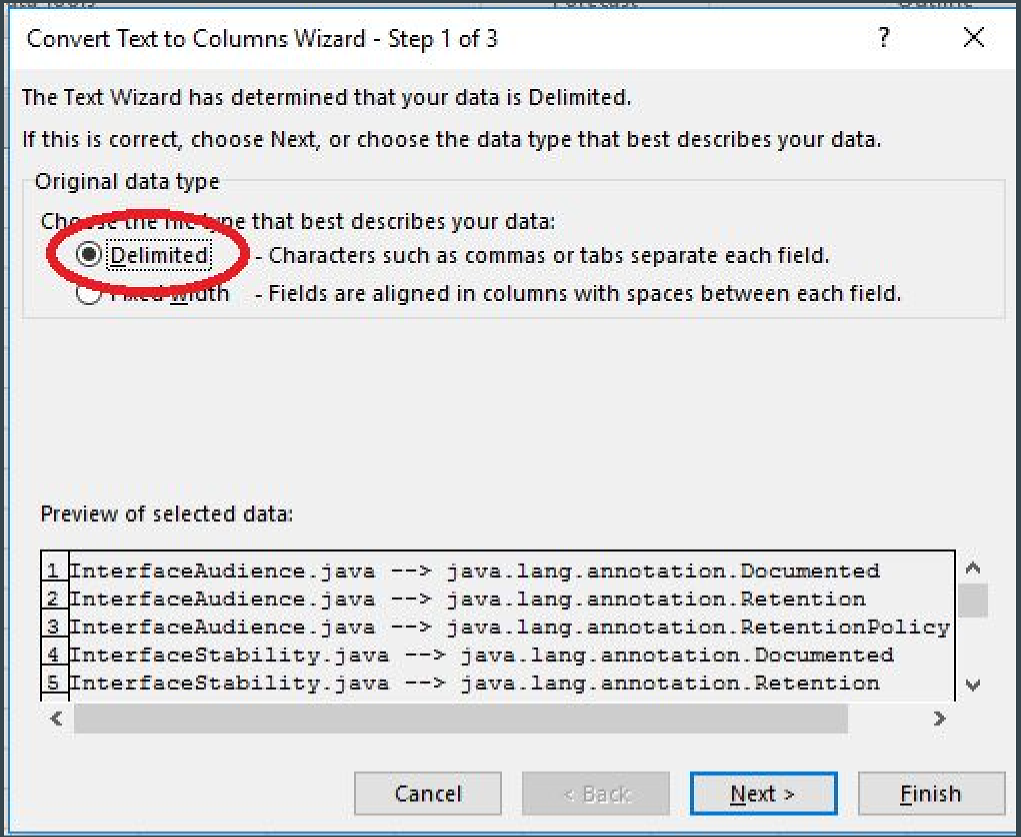
For example, you can separate a column of full names into separate first and last name columns.

You can choose how to split it up: fixed width or split at each comma, period, or other character.

Tell me more

	A
1	InterfaceAudience.java --> java.lang.annotation.Documented
2	InterfaceAudience.java --> java.lang.annotation.Retention
3	InterfaceAudience.java --> java.lang.annotation.RetentionPolicy
4	InterfaceStability.java --> java.lang.annotation.Documented
5	InterfaceStability.java --> java.lang.annotation.Retention
6	InterfaceStability.java --> java.lang.annotation.RetentionPolicy
7	InterfaceStability.java --> org.apache.hadoop.classification.InterfaceAudience.LimitedPrivate
8	InterfaceStability.java --> org.apache.hadoop.classification.InterfaceAudience.Private
9	InterfaceStability.java --> org.apache.hadoop.classification.InterfaceAudience.Public
10	ExcludePrivateAnnotationsJDiffDoclet.java --> com.sun.javadoc.DocErrorReporter
11	ExcludePrivateAnnotationsJDiffDoclet.java --> com.sun.javadoc.LanguageVersion
12	ExcludePrivateAnnotationsJDiffDoclet.java --> com.sun.javadoc.RootDoc
13	ExcludePrivateAnnotationsJDiffDoclet.java --> jdiff.JDiff
14	ExcludePrivateAnnotationsStandardDoclet.java --> com.sun.javadoc.DocErrorReporter
15	ExcludePrivateAnnotationsStandardDoclet.java --> com.sun.javadoc.LanguageVersion
16	ExcludePrivateAnnotationsStandardDoclet.java --> com.sun.javadoc.RootDoc
17	ExcludePrivateAnnotationsStandardDoclet.java --> com.sun.tools.doclets.standard.Standard
18	IncludePublicAnnotationsStandardDoclet.java --> com.sun.javadoc.DocErrorReporter
19	IncludePublicAnnotationsStandardDoclet.java --> com.sun.javadoc.LanguageVersion
20	IncludePublicAnnotationsStandardDoclet.java --> com.sun.javadoc.RootDoc
21	IncludePublicAnnotationsStandardDoclet.java --> com.sun.tools.doclets.standard.Standard
22	package-info.java --> org.apache.hadoop.classification.InterfaceAudience
23	RootDocProcessor.java --> com.sun.javadoc.AnnotationDesc
24	RootDocProcessor.java --> com.sun.javadoc.AnnotationTypeDoc
25	RootDocProcessor.java --> com.sun.javadoc.ClassDoc
26	RootDocProcessor.java --> com.sun.javadoc.ConstructorDoc
27	RootDocProcessor.java --> com.sun.javadoc.Doc
28	RootDocProcessor.java --> com.sun.javadoc.FieldDoc
29	RootDocProcessor.java --> com.sun.javadoc.MethodDoc
30	RootDocProcessor.java --> com.sun.javadoc.PackageDoc
31	RootDocProcessor.java --> com.sun.javadoc.ProgramElementDoc
32	RootDocProcessor.java --> com.sun.javadoc.RootDoc
33	RootDocProcessor.java --> java.lang.reflect.Array
34	RootDocProcessor.java --> java.lang.reflect.InvocationHandler
35	RootDocProcessor.java --> java.lang.reflect.InvocationTargetException
36	RootDocProcessor.java --> java.lang.reflect.Method
37	RootDocProcessor.java --> java.lang.reflect.Proxy
38	RootDocProcessor.java --> java.util.ArrayList

Comparison - Process



Convert Text to Columns Wizard - Step 1 of 3

The Text Wizard has determined that your data is Delimited.

If this is correct, choose Next, or choose the data type that best describes your data.

Original data type

Choose the file type that best describes your data:

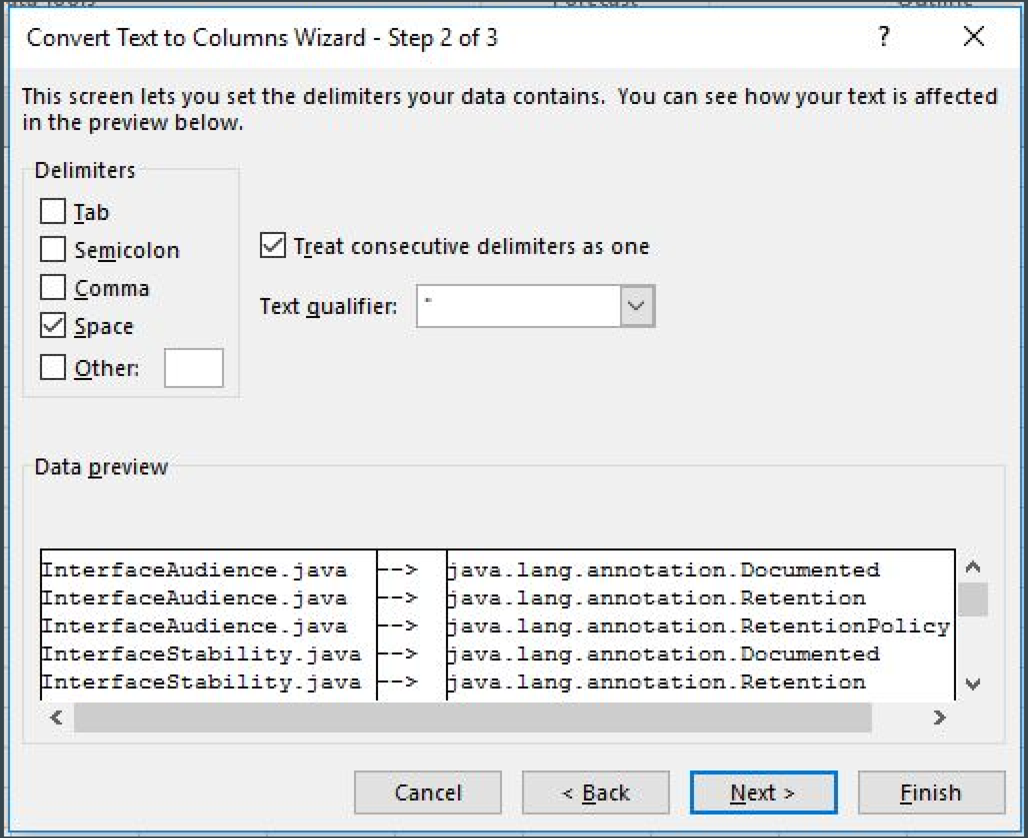
[✓ Delimited] - Characters such as commas or tabs separate each field.

Fixed width - Fields are aligned in columns with spaces between each field.

Preview of selected data:

```
1 InterfaceAudience.java --> java.lang.annotation.Documented
2 InterfaceAudience.java --> java.lang.annotation.Retention
3 InterfaceAudience.java --> java.lang.annotation.RetentionPolicy
4 InterfaceStability.java --> java.lang.annotation.Documented
5 InterfaceStability.java --> java.lang.annotation.Retention
```

Comparison - Process


Convert Text to Columns Wizard - Step 2 of 3

This screen lets you set the delimiters your data contains. You can see how your text is affected in the preview below.

Delimiters

- [ ] Tab
- [ ] Semicolon
- [ ] Comma
- [✓] Space
- [ ] Other:

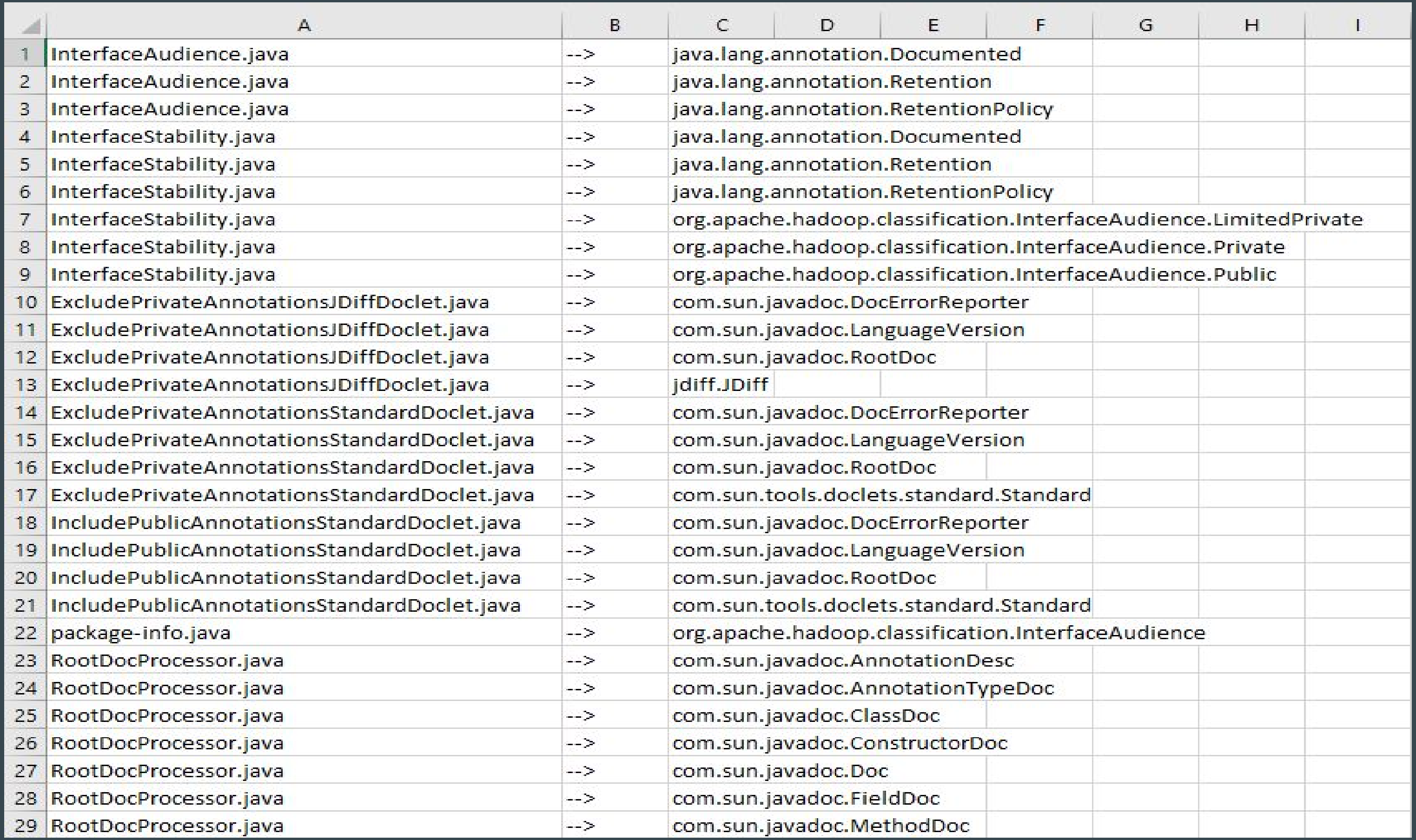
[✓] Treat consecutive delimiters as one

Text qualifier: "

Data preview

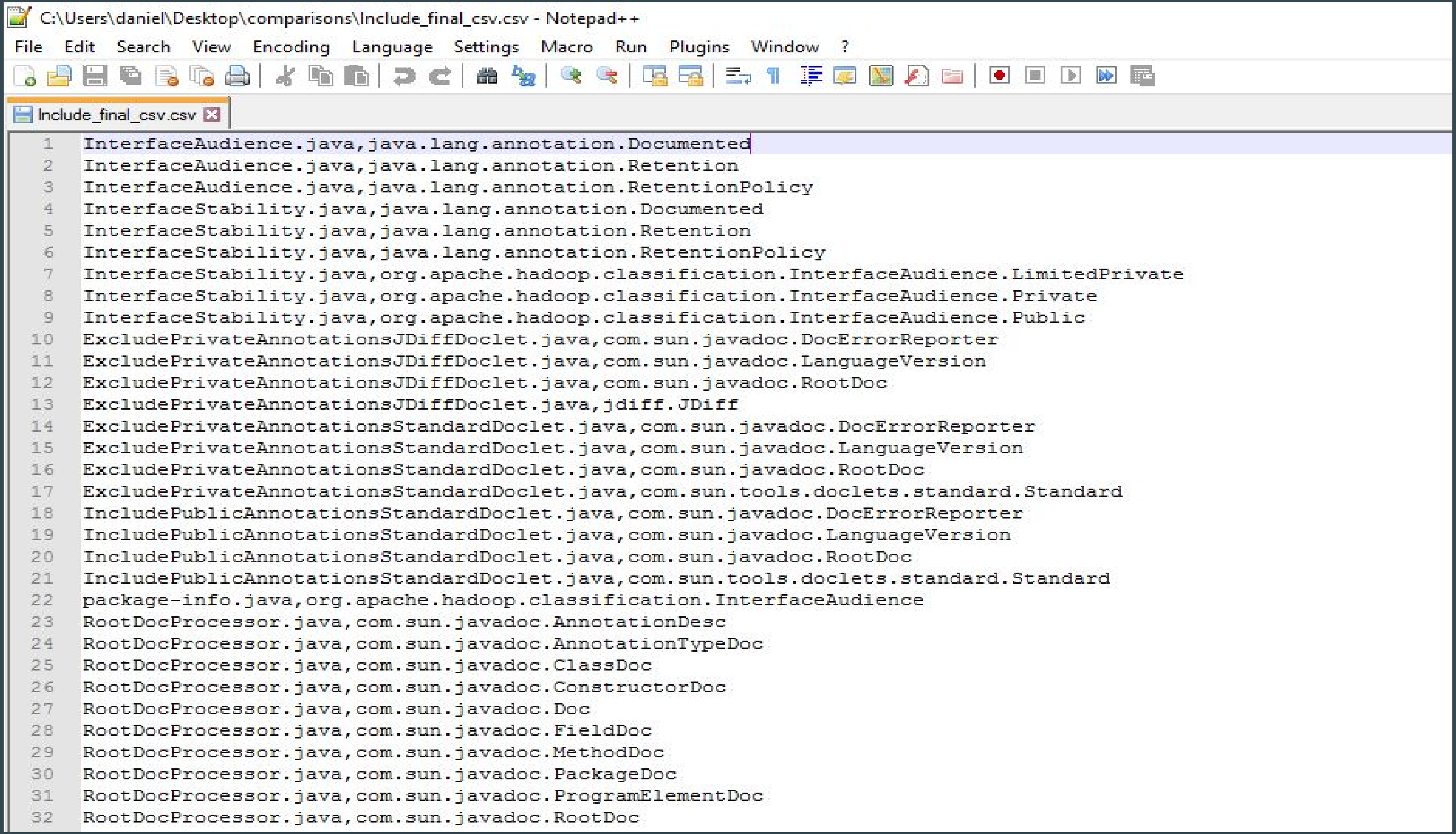
```
InterfaceAudience.java  -->  java.lang.annotation.Documented
InterfaceAudience.java  -->  java.lang.annotation.Retention
InterfaceAudience.java  -->  java.lang.annotation.RetentionPolicy
InterfaceStability.java -->  java.lang.annotation.Documented
InterfaceStability.java -->  java.lang.annotation.Retention
```

Comparison - Process



	A	B	C	D	E	F	G	H	I
1	InterfaceAudience.java	-->	java.lang.annotation.Documented						
2	InterfaceAudience.java	-->	java.lang.annotation.Retention						
3	InterfaceAudience.java	-->	java.lang.annotation.RetentionPolicy						
4	InterfaceStability.java	-->	java.lang.annotation.Documented						
5	InterfaceStability.java	-->	java.lang.annotation.Retention						
6	InterfaceStability.java	-->	java.lang.annotation.RetentionPolicy						
7	InterfaceStability.java	-->	org.apache.hadoop.classification.InterfaceAudience.LimitedPrivate						
8	InterfaceStability.java	-->	org.apache.hadoop.classification.InterfaceAudience.Private						
9	InterfaceStability.java	-->	org.apache.hadoop.classification.InterfaceAudience.Public						
10	ExcludePrivateAnnotationsJDiffDoclet.java	-->	com.sun.javadoc.DocErrorReporter						
11	ExcludePrivateAnnotationsJDiffDoclet.java	-->	com.sun.javadoc.LanguageVersion						
12	ExcludePrivateAnnotationsJDiffDoclet.java	-->	com.sun.javadoc.RootDoc						
13	ExcludePrivateAnnotationsJDiffDoclet.java	-->	jdiff.JDiff						
14	ExcludePrivateAnnotationsStandardDoclet.java	-->	com.sun.javadoc.DocErrorReporter						
15	ExcludePrivateAnnotationsStandardDoclet.java	-->	com.sun.javadoc.LanguageVersion						
16	ExcludePrivateAnnotationsStandardDoclet.java	-->	com.sun.javadoc.RootDoc						
17	ExcludePrivateAnnotationsStandardDoclet.java	-->	com.sun.tools.doclets.standard.Standard						
18	IncludePublicAnnotationsStandardDoclet.java	-->	com.sun.javadoc.DocErrorReporter						
19	IncludePublicAnnotationsStandardDoclet.java	-->	com.sun.javadoc.LanguageVersion						
20	IncludePublicAnnotationsStandardDoclet.java	-->	com.sun.javadoc.RootDoc						
21	IncludePublicAnnotationsStandardDoclet.java	-->	com.sun.tools.doclets.standard.Standard						
22	package-info.java	-->	org.apache.hadoop.classification.InterfaceAudience						
23	RootDocProcessor.java	-->	com.sun.javadoc.AnnotationDesc						
24	RootDocProcessor.java	-->	com.sun.javadoc.AnnotationTypeDoc						
25	RootDocProcessor.java	-->	com.sun.javadoc.ClassDoc						
26	RootDocProcessor.java	-->	com.sun.javadoc.ConstructorDoc						
27	RootDocProcessor.java	-->	com.sun.javadoc.Doc						
28	RootDocProcessor.java	-->	com.sun.javadoc.FieldDoc						
29	RootDocProcessor.java	-->	com.sun.javadoc.MethodDoc						

Comparison - Process


C:\Users\daniel\Desktop\comparisons\Include_final_csv.csv - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
Include_final_csv.csv

```
InterfaceAudience.java,java.lang.annotation.Documented
InterfaceAudience.java,java.lang.annotation.Retention
InterfaceAudience.java,java.lang.annotation.RetentionPolicy
InterfaceStability.java,java.lang.annotation.Documented
InterfaceStability.java,java.lang.annotation.Retention
InterfaceStability.java,java.lang.annotation.RetentionPolicy
InterfaceStability.java,org.apache.hadoop.classification.InterfaceAudience.LimitedPrivate
InterfaceStability.java,org.apache.hadoop.classification.InterfaceAudience.Private
InterfaceStability.java,org.apache.hadoop.classification.InterfaceAudience.Public
ExcludePrivateAnnotationsJDiffDoclet.java,com.sun.javadoc.DocErrorReporter
ExcludePrivateAnnotationsJDiffDoclet.java,com.sun.javadoc.LanguageVersion
ExcludePrivateAnnotationsJDiffDoclet.java,com.sun.javadoc.RootDoc
ExcludePrivateAnnotationsJDiffDoclet.java,jdiff.JDiff
ExcludePrivateAnnotationsStandardDoclet.java,com.sun.javadoc.DocErrorReporter
ExcludePrivateAnnotationsStandardDoclet.java,com.sun.javadoc.LanguageVersion
ExcludePrivateAnnotationsStandardDoclet.java,com.sun.javadoc.RootDoc
ExcludePrivateAnnotationsStandardDoclet.java,com.sun.tools.doclets.standard.Standard
IncludePublicAnnotationsStandardDoclet.java,com.sun.javadoc.DocErrorReporter
IncludePublicAnnotationsStandardDoclet.java,com.sun.javadoc.LanguageVersion
IncludePublicAnnotationsStandardDoclet.java,com.sun.javadoc.RootDoc
IncludePublicAnnotationsStandardDoclet.java,com.sun.tools.doclets.standard.Standard
package-info.java,org.apache.hadoop.classification.InterfaceAudience
RootDocProcessor.java,com.sun.javadoc.AnnotationDesc
RootDocProcessor.java,com.sun.javadoc.AnnotationTypeDoc
RootDocProcessor.java,com.sun.javadoc.ClassDoc
RootDocProcessor.java,com.sun.javadoc.ConstructorDoc
RootDocProcessor.java,com.sun.javadoc.Doc
RootDocProcessor.java,com.sun.javadoc.FieldDoc
RootDocProcessor.java,com.sun.javadoc.MethodDoc
RootDocProcessor.java,com.sun.javadoc.PackageDoc
RootDocProcessor.java,com.sun.javadoc.ProgramElementDoc
RootDocProcessor.java,com.sun.javadoc.RootDoc
```

Comparison - Qualitative Results

Comparison - Qualitative Results

- Precision and Recall
    - Precision (Positive Predictive Value) is the fraction of retrieved instances that are relevant
    - Recall (Sensitivity) is the fraction of relevant instances that are retrieved
- Based on an understanding and measurement of relevancy

Comparison - Qualitative Results

- INCLUDE Precision and Recall
    - Precision:

83964 total dependencies
79015 dependencies identified
1849 unique (correct) dependencies identified

1849/79015 = ***0.002340062013***

    - Recall:

83964 total dependencies
79015 dependencies identified
1849 unique (correct) dependencies identified

1849/83964 = ***0.02202134248***

Comparison - Qualitative Results

- SRCML Precision and Recall
    - Precision:

83964 total dependencies
82115 dependencies identified
4949 unique (correct) dependencies identified

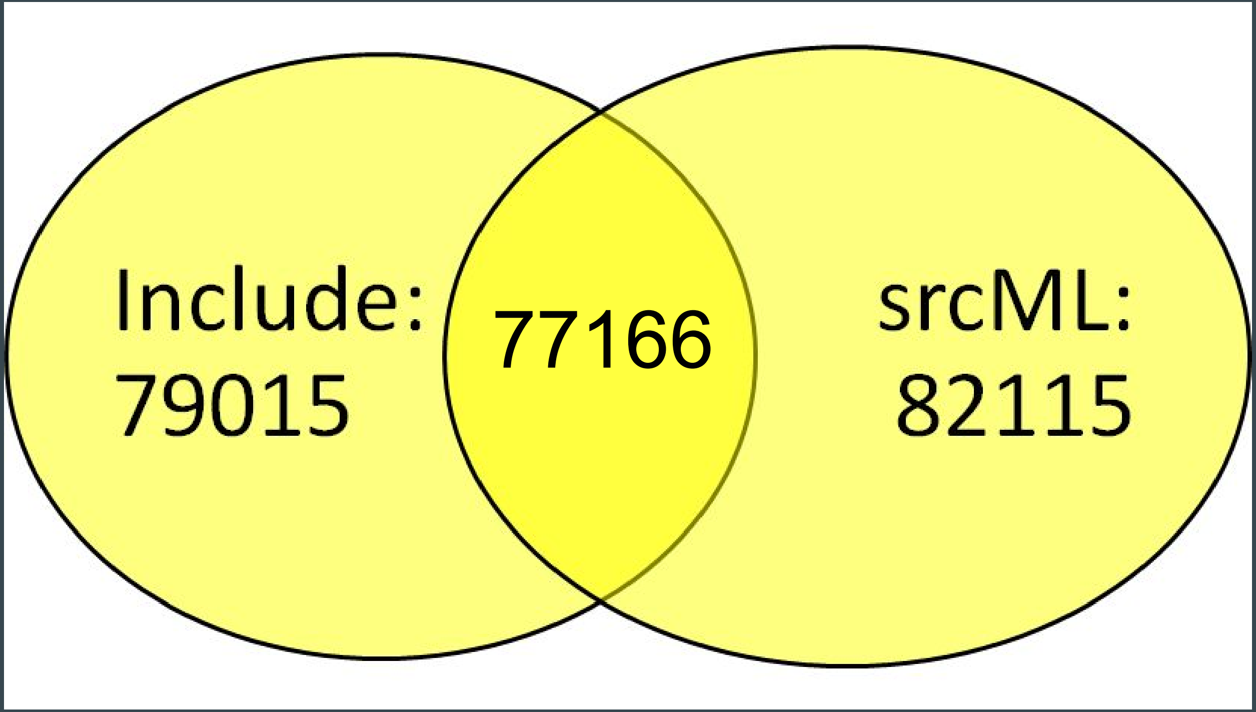
4949/82115 = ***0.06026913475***

    - Recall:

83964 total dependencies
82115 dependencies identified
4949 unique (correct) dependencies identified

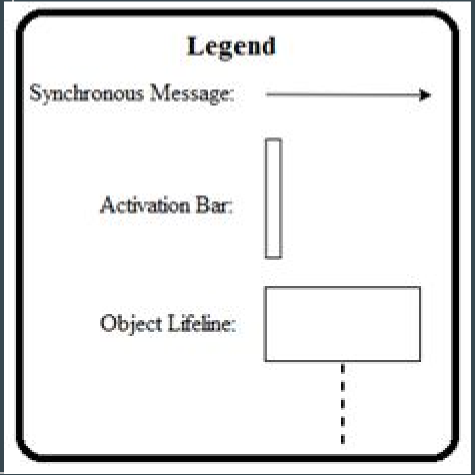
4949/83964 = ***0.05894192749***

Comparison - Qualitative Results

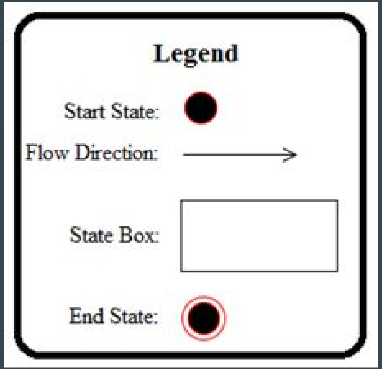


TOTAL: 83964

Use Case: Extracting Dependencies with Understand



Use Case: Extracting Dependencies with SrcML



Limitations of Reported Findings

- Since a random sampling was taken, the comparison may not be representative of the entire system
- The software used might not have been able to hold the full list of dependencies and therefore a small amount may have been lost as a result
- Since the system is large and complex, each technique found a different list of dependencies (however many similar)

Lessons Learned

- There are many ways to extract the dependencies of a system
- There are many tools to help in the process of dependency extraction
- Dealing with large lists of data may present some unexpected problems
- Do not brute force your way through an entire system!
    - Use the sample size calculator!

Conclusion

- Although there are many techniques to extract the dependencies of a system, they create different results
- In a system so complex, it is unrealistic that two dependency extraction techniques yield exactly the same results