Lab3 实验报告

1. 思路概述

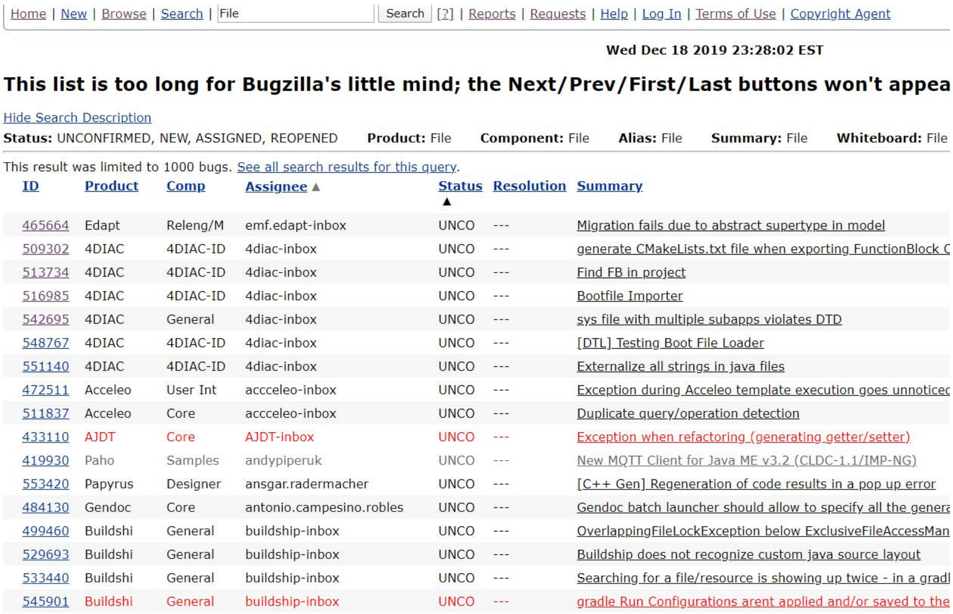
本次实验我们小组仍以 Eclipse 官网提供的 bug 缺陷报告数据作为原材料，将这些 bug 条目视为各种需求，进行分析。

确定软件需求是一个不简单的步骤，因为 IDE 的需求繁多庞杂，涉及到极多方面的需求或者缺陷。因此我们首先人工讨论和初步查看之后，先决定一个大方向的需求，然后利用爬虫技术将相关的需求信息获取下来，再经过分析确定最终的软件需求 R，从该需求开始进行实验的后续步骤

2. 明确需求 R

首先要通过爬虫技术来进行需求的筛选，因为很多缺陷报告都是阐述单一的缺陷和错误，没有变更历史和后续跟进，这种需求是难以进行实验的，无法进行需求变化的跟踪。基于这个思路，我们必须先筛选出包含一些历史改进记录或者持续更新的缺陷报告。

我们小组决定以’文件’(File)相关的需求进行入手，由于实验 2 中已经对 eclipse 官网进行过分析并阐述过设计爬虫的思路，这里只需稍作改进即可，首先构造搜索目标 url：https://bugs.eclipse.org/bugs/buglist.cgi?quicksearch=File，进行网址请求之后得到如下的网页源码。再获取第一列的所有 ID 号，进行下一步的爬取



Home | New | Browse | Search | File Search | [?] | Reports | Requests | Help | Log In | Terms of Use | Copyright Agent

Wed Dec 18 2019 23:28:02 EST

This list is too long for Bugzilla's little mind; the Next/Prev/First/Last buttons won't appea

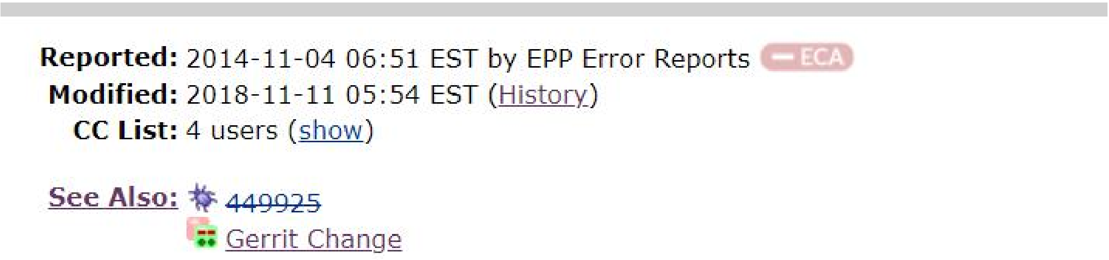
Hide Search Description

Status: UNCONFIRMED, NEW, ASSIGNED, REOPENED **Product:** File **Component:** File **Alias:** File **Summary:** File **Whiteboard:** File

This result was limited to 1000 bugs. See all search results for this query.

ID	Product	Comp	Assignee ▲	Status ▲	Resolution	Summary
465664	Edapt	Releng/M	emf.edapt-inbox	UNCO	---	Migration fails due to abstract supertype in model
509302	4DIAC	4DIAC-ID	4diac-inbox	UNCO	---	generate CMakeLists.txt file when exporting FunctionBlock C
513734	4DIAC	4DIAC-ID	4diac-inbox	UNCO	---	Find FB in project
516985	4DIAC	4DIAC-ID	4diac-inbox	UNCO	---	Bootfile Importer
542695	4DIAC	General	4diac-inbox	UNCO	---	sys file with multiple subapps violates DTD
548767	4DIAC	4DIAC-ID	4diac-inbox	UNCO	---	[DTL] Testing Boot File Loader
551140	4DIAC	4DIAC-ID	4diac-inbox	UNCO	---	Externalize all strings in java files
472511	Acceleo	User Int	accceleo-inbox	UNCO	---	Exception during Acceleo template execution goes unnoticec
511837	Acceleo	Core	accceleo-inbox	UNCO	---	Duplicate query/operation detection
433110	AJDT	Core	AJDT-inbox	UNCO	---	Exception when refactoring (generating getter/setter)
419930	Paho	Samples	andypiperuk	UNCO	---	New MQTT Client for Java ME v3.2 (CLDC-1.1/IMP-NG)
553420	Papyrus	Designer	ansgar.radermacher	UNCO	---	[C++ Gen] Regeneration of code results in a pop up error
484130	Gendoc	Core	antonio.campesino.robles	UNCO	---	Gendoc batch launcher should allow to specify all the genera
499460	Buildshi	General	buildship-inbox	UNCO	---	OverlappingFileLockException below ExclusiveFileAccessMan
529693	Buildshi	General	buildship-inbox	UNCO	---	Buildship does not recognize custom java source layout
533440	Buildshi	General	buildship-inbox	UNCO	---	Searching for a file/resource is showing up twice - in a gradl
545901	Buildshi	General	buildship-inbox	UNCO	---	gradle Run Configurations arent applied and/or saved to the

图 2-1 搜索‘File’的大量结果报告



Reported: 2014-11-04 06:51 EST by EPP Error Reports ECA
Modified: 2018-11-11 05:54 EST (History)
CC List: 4 users (show)

See Also: ~~449925~~
Gerrit Change

图 2-2 某报告中’See Also’字段

进入到某个 ID 号对应的报告中，可以发现在第二列的’See Also’字段中，有一些 commit 的记录和被划掉的 ID 号，经过分析之后这些被划掉的 ID 号对应的 bug 和当前 ID 的 bug 是有联系和前导依赖的，这也恰好符合实验所需要的需求跟踪分析目的，因此设计的爬虫可以进行网页追踪，具体步骤如下：

（1）对于每一个 ID，首先创建一个跟踪的 ID 集合，初始化元素只有这个 ID

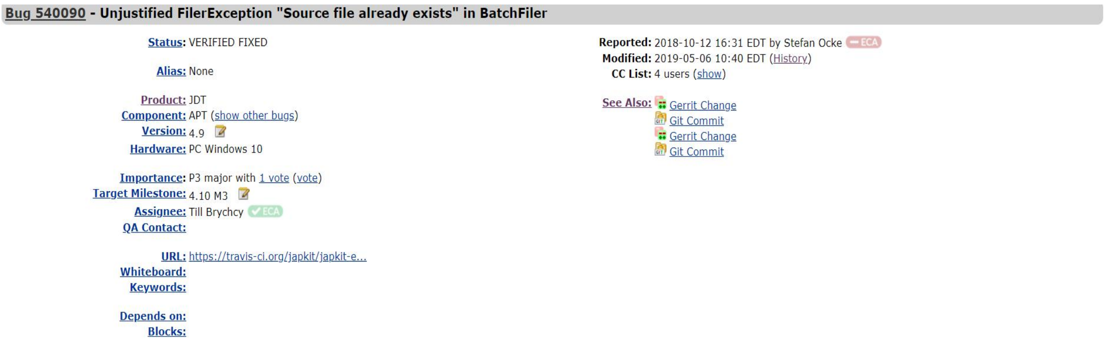
（2）进入这个 ID 对应的报告，观察‘See Also’字段是否具有相关的 ID 号和 commit 记录

（3）如果没有相关的 ID，则这个 ID 对应的 bug/需求追踪结束。

（4）如果有，则获取这些相关的 ID，加入到集合中，并且递归地对这些 ID 进行请求，在新的报告中观察是否继续有相关 ID，依次类推，直到追踪到尽头

（5）结束追踪之后，如果集合的元素仍为 1，则丢弃这个 ID 对应的需求，继续下一个，否则将这组 ID 集合加到最终输出文本中

利用这个思路设计的爬虫，经过筛选之后所剩的集合元素并不多，我们小组经过分析之后找到一个比较易懂且符合我们期望的需求，最终的需求/bug 为：**540090- Unjustified FilerException "Source file already exists" in BatchFiler**。严格地说它是一个发生了改动变化的 bug 而不算是真正的改进的需求。这个需求主要描述的问题：“某些情况下出现了“源文件已创建”的不合理异常提示“



Bug 540090 - Unjustified FilerException "Source file already exists" in BatchFiler

Status: VERIFIED FIXED

Alias: None

Product: JDT
Component: APT (show other bugs)
Version: 4.9
Hardware: PC Windows 10

Importance: P3 major with 1 vote (vote)
Target Milestone: 4.10 M3
Assignee: Till Brychcy ✓ECA
QA Contact:

URL: https://travis-ci.org/japkit/japkit-e...
Whiteboard:
Keywords:

Depends on:
Blocks:

Reported: 2018-10-12 16:31 EDT by Stefan Ocke ECA
Modified: 2019-05-06 10:40 EDT (History)
CC List: 4 users (show)

See Also: Gerrit Change
Git Commit
Gerrit Change
Git Commit

图 2-3 需求 R 的缺陷报告

注意到它还有几个需求有依赖相关，以及几次代码更改的 commit，接下来将会将这些变更信息进行提取和整理

3.需求实现代码

```
public class IdeFilerImpl implements Filer {

            //private final IdeAnnotationProcessorManager _dispatchManager;
            private final IdeProcessingEnvImpl _env;

            public IdeFilerImpl(IdeAnnotationProcessorManager dispatchManager,
                                    IdeProcessingEnvImpl env) {
                        //_dispatchManager = dispatchManager;
                        _env = env;
            }

            /* (non-Javadoc)
             * @see javax.annotation.processing.Filer#createClassFile(java.lang.CharSequence, javax.lang.model.element.Element[])
             */
            @Override
            public JavaFileObject createClassFile(CharSequence name, Element... originatingElements)
                                    throws IOException {

                        // Pre-emptively check parameters here, rather than later on when the resource is written and closed.
                        if (null == name) {
                                    throw new IllegalArgumentException("Name is null");
                        }

            String nameAsString = name.toString();
                        IFile file = _env.getAptProject().getGeneratedFileManager(_env.isTestCode()).getIFileForTypeName(nameAsString);

            GeneratedSourceFolderManager gsfm = _env.getAptProject().getGeneratedSourceFolderManager(_env.isTestCode());
            IPath path = null;
            try {
                                    path = gsfm.getBinaryOutputLocation();
                        } catch (JavaModelException e) {
                                    Apt6Plugin.log(e, "Failure getting the binary output location"); // $NON-NLS-1$
                                    throw new IOException(e);
                        }
            int index = nameAsString.lastIndexOf('.');
            String pkg = null;
            if (index != -1) {
                        name = nameAsString.substring(index + 1);
                        pkg = nameAsString.substring(0, index);
            } else {
                        pkg = "";
            }
            file = getFileFromOutputLocation(StandardLocation.CLASS_OUTPUT, pkg, name + ".class");
                        path = path.append(nameAsString);
                        path = new Path(path.toString() + ".class");

                        return new IdeOutputClassFileObject(_env, file, nameAsString);
            }

            /* (non-Javadoc)
             * @see javax.annotation.processing.Filer#createResource(javax.tools.JavaFileManager.Loca
tion, java.lang.CharSequence, java.lang.CharSequence, javax.lang.model.element.Element[])
             * In the IDE implementation, we support only two Locations: SOURCE_OUTPUT, which means th
e APT generated source folder,
             * and CLASS_OUTPUT, which means the binary output folder associated with the APT generate
d source folder.
             */
            @Override
            public FileObject createResource(Location location, CharSequence pkg,
                                    CharSequence relativeName, Element... originatingElements) throws I
OException
            {
                        // Pre-emptively check parameters here, rather than later on when the resource
is written and closed.
                        if (null == location) {
                                    throw new IllegalArgumentException("Location is null");
                        }
                        if (!location.isOutputLocation()) {
                                    throw new IllegalArgumentException("Location " + location.getName()
 + " is not an output location");
                        }

                        if (null == pkg) {
                                    throw new IllegalArgumentException("Package is null");
                        }
                        if (null == relativeName) {
                                    throw new IllegalArgumentException("Relative name is null");
                        }
```

```
        if ( relativeName.length() == 0) {
            throw new IllegalArgumentException("Relative name is zero length");
        }
        IFile file = getFileFromOutputLocation(location, pkg, relativeName);
        if (AptCompilationParticipant.getInstance().getJava6GeneratedFiles().contains
(file)) {
            throw new FilerException("Source file already created: " + file.getF
ullPath()); //$NON-NLS-1$
        }

        Set<IFile> parentFiles;
        if (originatingElements != null && originatingElements.length > 0) {
            parentFiles = new HashSet<IFile>(originatingElements.length);
            for (Element elem : originatingElements) {
                IFile enclosing = _env.getEnclosingIFile(elem);
                if (null != enclosing) {
                    parentFiles.add(enclosing);
                }
            }
        } else {
            parentFiles = Collections.emptySet();
        }
        return new IdeOutputNonSourceFileObject(_env, file, parentFiles);
    }

    /**
     * @param originatingElements should all be source types; binary types (ie elements in jar
 files)
     * will be ignored.
     * @see javax.annotation.processing.Filer#createSourceFile(java.lang.CharSequence, javax.
lang.model.element.Element[])
     */
    @Override
    public JavaFileObject createSourceFile(CharSequence name, Element... originatingElements)
            throws IOException
    {
        // Pre-emptively check parameters here, rather than later on when the resource
is written and closed.
        if (null == name) {
            throw new IllegalArgumentException("Name is null");
        }
        IFile file = _env.getAptProject().getGeneratedFileManager(_env.isTestCode()).g
etIFileForTypeName(name.toString());
```

```
                    if (AptCompilationParticipant.getInstance().getJava6GeneratedFiles().contains
(file)) {
                              throw new FilerException("Source file already created: " + file.getF
ullPath()); //$NON-NLS-1$
                    }
                    // TODO: is the following correct?
                    // JDK 9's createSourceFile API mentions '/' as separator for a module prefix.
                    // Otherwise shouldn't <code>name</code> already be "."-separated?
                    name = name.toString().replace('/', '.');

                    ModuleBinding m = _env._current_module;
                    if (m == null)
                              m = _env.getCompiler().lookupEnvironment.UnNamedModule;
                    ReferenceBinding type = m.environment.getType(CharOperation.splitOn('.', name.
toString().toCharArray()), m);
                    if (type != null && (type.tagBits & TagBits.HasMissingType) == 0) {
                              throw new FilerException("Source file already exists : " + name); //
$NON-NLS-1$
                    }
                    Set<IFile> parentFiles = Collections.emptySet();
                    if (originatingElements != null && originatingElements.length > 0) {
                              parentFiles = new HashSet<IFile>(originatingElements.length);
                              for (Element elem : originatingElements) {
                                        IFile enclosing = _env.getEnclosingIFile(elem);
                                        if (null != enclosing) {
                                                  parentFiles.add(enclosing);
                                        }
                              }
                    }
                    return new IdeOutputJavaFileObject(_env, name, parentFiles);
          }

          /* (non-Javadoc)
           * @see javax.annotation.processing.Filer#getResource(javax.tools.JavaFileManager.Locatio
n, java.lang.CharSequence, java.lang.CharSequence)
           * Returns a FileObject representing the specified resource.  The only supported locations
           * are CLASS_OUTPUT and SOURCE_OUTPUT.
           */
          @Override
          public FileObject getResource(Location location, CharSequence pkg, CharSequence relativeNa
me)
                              throws IOException
          {
                    IFile file = getFileFromOutputLocation(location, pkg, relativeName);
```

```
                      return new IdeInputFileObject(file);
           }

    /**
     * Return a project-relative path.  This does not create the file nor its parent directories,
     * but it does validate the path.
     * @param pkg must be non-null but can be empty.
     * @param relPath must be non-null and non-empty.
     * @throws IOException if the path is not valid.
     */
    protected IFile getFileFromOutputLocation( Location loc, CharSequence pkg, CharSequence relPath )
           throws IOException
    {
           GeneratedSourceFolderManager gsfm = _env.getAptProject().getGeneratedSourceFolderManager
(_env.isTestCode());
           IPath path = null;
           if ( loc == StandardLocation.CLASS_OUTPUT )
           {
                      try
                      {
                                 path = gsfm.getBinaryOutputLocation();
                      }
                      catch ( JavaModelException e )
                      {
                                 Apt6Plugin.log(e, "Failure getting the binary output location"); //
$NON-NLS-1$
                                 throw new IOException(e);
                      }
           }
           else if ( loc == StandardLocation.SOURCE_OUTPUT ) {
                      path = gsfm.getFolder().getProjectRelativePath();
           }
           else {
                      throw new IllegalArgumentException("Unsupported location: " + loc);
           }

        if( pkg.length() > 0 )
            path = path.append(pkg.toString().replace('.', File.separatorChar) );

        path = path.append(relPath.toString());

        IFile file = _env.getProject().getFile(path);

                      validatePath(file);
            return file;
    }

    /**
     * Validate that a path fits the rules for being created.
     * @see IWorkspace#validatePath()
     * @throws IOException
     */
    private void validatePath(IFile file) throws IOException
            {
            IStatus status = _env.getProject().getWorkspace().validatePath(file.getFullPath().toOSStr
ing(), IResource.FILE);
            if (!status.isOK()) {
            CoreException ce = new CoreException(status);
            throw new IOException("Invalid path: " + file.toString(), ce); //$NON-NLS-1$
        }
            }

}
```

这部分代码来自 IdeFilerImpl.java，主要是实现了我们确定的需求 R(不合理地弹出了"源文件已创建"的提示)，而这个异常实际上是 174 行的 **throw new FilerException("Source file already exists : " + name)** 这个异常抛出导致的，所以提交此需求的相关人员对此做出了一些变更

相关 commit 链接：

https://git.eclipse.org/c/jdt/eclipse.jdt.core.git/commit/?id=d33fe2c4fa66e382227d407614e9d343a136e103

https://git.eclipse.org/c/jdt/eclipse.jdt.core.git/commit/?id=d33fe2c4fa66e382227d407614e9d343a136e103

3. 需求变更

(1) 变更讨论文本

· Dirk Ziegenbalg CLA Friend 2018-12-04 09:02:45 EST

I'm using mapstruct as annotation-processor to generate mappers for mapping entities to dtos and vice versa. With M3-Milestone-Release I'm getting a FilerException "Source file already created" when editing a mapper and than saving this file. Now the autobuild is going on and I'm getting the error. When I now call clean projects then the error goes away and the source file is generated.

I'd setup and attach a small workspace where the error occurs.

Comment #18 of bug 367599 (https://bugs.eclipse.org/bugs/show_bug.cgi?id=367599#c18) seems to be the same problem with another anntotation processor.

·Dani Megert CLA Friend 2018-12-04 10:14:51 EST

Does it work with M2? If not, did it work with 4.9?

So, you have an ugly workaround?

·Till Brychcy CLA Friend 2018-12-04 11:12:31 EST

Thanks, I can reproduce.

I think the whole approach of just checking whether the type exists is broken because of incremental compilation.

My patch from bug Bug 539774 just improved the situation w.r.t. change from Bug 534979, but I think we should just disable this check for 4.10RC2

·Till Brychcy CLA Friend 2018-12-05 12:10:41 EST

I think I was not clear: what about files that are regularly created only in a later round (e.g. because an annotation processor is applied to a generated file from the previous round).

Wouldn't the type for these still exist during incremental compilation and an error be thrown?

（2）变更代码

```
+@SupportedAnnotationTypes("targets.AnnotationProcessorTests.Bug540090.Bug540090")
+@SupportedSourceVersion(SourceVersion.RELEASE_6)
+public class Bug540090Proc extends AbstractProcessor
+{
+ @Override
+ public boolean process(Set<? extends TypeElement> annotations,
+ RoundEnvironment roundEnv) {
+ Filer filer = processingEnv.getFiler();
+ if (!annotations.isEmpty()) {
+ try {
+ JavaFileObject jfo = filer.createSourceFile("gen.GenClass");
+ Writer writer = jfo.openWriter();
+ writer.write("package gen;\n");
+ writer.write("public class GenClass {\n");
+ writer.write(" int x;");
+ writer.write("}\n");
+ writer.close();
+ return true;
+ } catch (IOException e) {
+ return false;
+ }
+ }
+ return true;
+ }
+}
```

```
+package targets.AnnotationProcessorTests.Bug540090;
+
+@Bug540090
+public class B {
+ public void foo(gen.GenClass x) {
+ return;
+ }
+}
```

```
+package targets.AnnotationProcessorTests.Bug540090;
+
+public @interface Bug540090 {}
```

```
- * Copyright (c) 2006, 2017 IBM, BEA Systems, Inc. and others
+ * Copyright (c) 2006, 2018 IBM, BEA Systems, Inc. and others
  *
  * This program and the accompanying materials
  * are made available under the terms of the Eclipse Public License 2.0
@@ -35,6 +35,7 @@ public class AllTests extends TestCase {
          suite.addTestSuite(NegativeTests.class);
          suite.addTestSuite(Java8ElementsTests.class);
          suite.addTestSuite(Java9ElementsTests.class);
+         suite.addTestSuite(Java11ElementsTests.class);
          suite.addTestSuite(AnnotationProcessorTests.class);
          return suite;
      }
```

```
- * Copyright (c) 2014, 2015 IBM Corporation and others.
+ * Copyright (c) 2014, 2018 IBM Corporation and others.
  *
  * This program and the accompanying materials
  * are made available under the terms of the Eclipse Public License 2.0
@@ -97,6 +97,20 @@ public class AnnotationProcessorTests extends TestCase {
          assertEquals(true, success);
      }

+     public void testBug540090() throws IOException {
```

```
+       JavaCompiler compiler = BatchTestUtils.getEclipseCompiler();
+       File targetFolder = TestUtils.concatPath(BatchTestUtils.getSrcFolderName(), "targets",
"AnnotationProcessorTests", "bug540090");
+       BatchTestUtils.copyResources("targets/AnnotationProcessorTests/bug540090",
targetFolder);
+       List<String> options = new ArrayList<String>();
+       final String PROC =
"org.eclipse.jdt.compiler.apt.tests.processors.AnnotationProcessorTests.Bug540090Proc";
+       options.add("-processorpath");
+       options.add(" ");
+       options.add("-processor");
+       options.add(PROC);
+       boolean success = BatchTestUtils.compileTreeWithErrors(compiler, options, targetFolder,
null);
+       assertEquals(true, success);
+   }
+
    public void testBug415274() throws IOException {
        JavaCompiler compiler = BatchTestUtils.getEclipseCompiler();
        File targetFolder = TestUtils.concatPath(BatchTestUtils.getSrcFolderName(), "targets",
"AnnotationProcessorTests", "bug415274");

- * Copyright (c) 2006, 2017 BEA Systems, Inc. and others
+ * Copyright (c) 2006, 2018 BEA Systems, Inc. and others
  *
  * This program and the accompanying materials
  * are made available under the terms of the Eclipse Public License 2.0
@@ -649,6 +649,9 @@ public class ElementsImpl implements Elements {
        if (null == binding) {
            return null;
        }
+       if((binding.tagBits & TagBits.HasMissingType) != 0) {
+           return null;
+       }
        return new TypeElementImpl(_env, binding, null);
    }
```

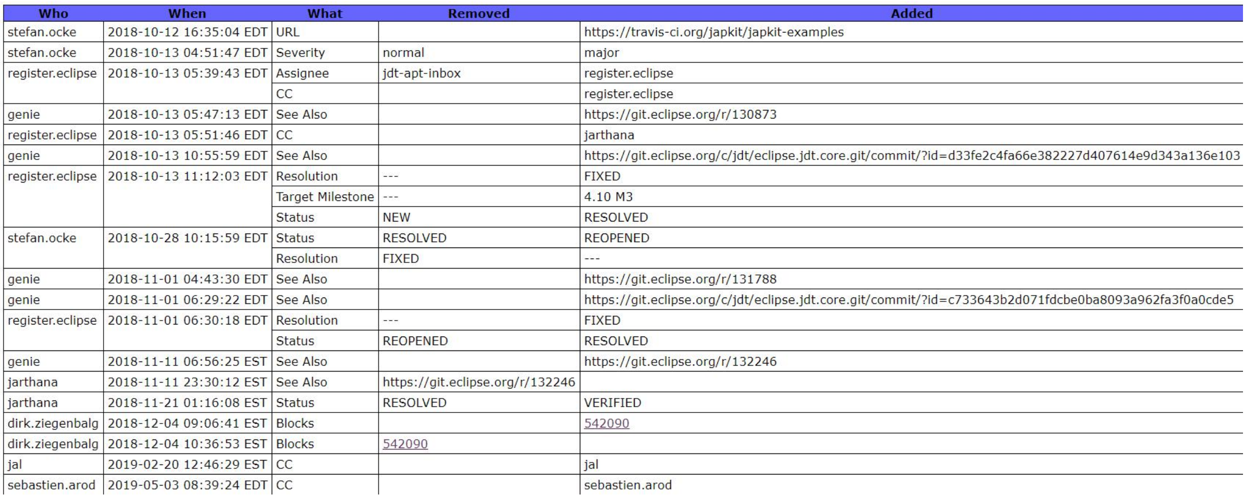
这段代码主要针对异常处理 throw new FilerException("Source file already exists : " + name); 进行了修改

相关 git commit 链接：

https://git.eclipse.org/c/jdt/eclipse.jdt.core.git/commit/?id=178dc273b0765dcfaf9cca6a214c89ecf4c18d31

https://git.eclipse.org/c/jdt/eclipse.jdt.core.git/commit/?id=d33fe2c4fa66e382227d407614 e9d343a136e103

4. 需求时间线

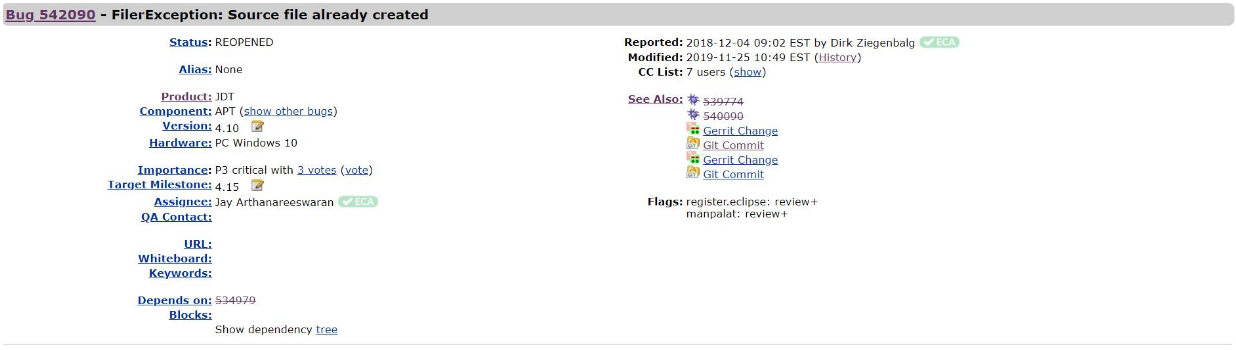


Who	When	What	Removed	Added
stefan.ocke	2018-10-12 16:35:04 EDT	URL		https://travis-ci.org/japkit/japkit-examples
stefan.ocke	2018-10-13 04:51:47 EDT	Severity	normal	major
register.eclipse	2018-10-13 05:39:43 EDT	Assignee	jdt-apt-inbox	register.eclipse
		CC		register.eclipse
genie	2018-10-13 05:47:13 EDT	See Also		https://git.eclipse.org/r/130873
register.eclipse	2018-10-13 05:51:46 EDT	CC		jarthana
genie	2018-10-13 10:55:59 EDT	See Also		https://git.eclipse.org/c/jdt/eclipse.jdt.core.git/commit/?id=d33fe2c4fa66e382227d407614e9d343a136e103
register.eclipse	2018-10-13 11:12:03 EDT	Resolution	---	FIXED
		Target Milestone	---	4.10 M3
		Status	NEW	RESOLVED
stefan.ocke	2018-10-28 10:15:59 EDT	Status	RESOLVED	REOPENED
		Resolution	FIXED	---
genie	2018-11-01 04:43:30 EDT	See Also		https://git.eclipse.org/r/131788
genie	2018-11-01 06:29:22 EDT	See Also		https://git.eclipse.org/c/jdt/eclipse.jdt.core.git/commit/?id=c733643b2d071fdcbe0ba8093a962fa3f0a0cde5
register.eclipse	2018-11-01 06:30:18 EDT	Resolution	---	FIXED
		Status	REOPENED	RESOLVED
genie	2018-11-11 06:56:25 EST	See Also		https://git.eclipse.org/r/132246
jarthana	2018-11-11 23:30:12 EST	See Also	https://git.eclipse.org/r/132246	
jarthana	2018-11-21 01:16:08 EST	Status	RESOLVED	VERIFIED
dirk.ziegenbalg	2018-12-04 09:06:41 EST	Blocks		542090
dirk.ziegenbalg	2018-12-04 10:36:53 EST	Blocks	542090	
jal	2019-02-20 12:46:29 EST	CC		jal
sebastien.arod	2019-05-03 08:39:24 EDT	CC		sebastien.arod

图 5-1 540090 号需求修改历史记录

为了获取需求变化的时间线和重要事件，我们获取了该缺陷报告中“Modified”字段的信息，里面呈现了一些需求变更的原材料和数据，并能沿着它所给的后续依赖找到更多的变更线索

注意到”FIXED--->RESOLVED--->REOPENED“,这三个关键字反映的是一次变更和改动的过程，而显然该 540090 号 bug 发生了两次变更，变更的代码文本和讨论文本已在上文中提及，不再详细阐述。而在验证正确之后，出现了新的后续依赖需求：542090，因此我们继续对 542090 号 bug 进行信息的获取



Bug 542090 - FilerException: Source file already created

Status: REOPENED

Alias: None

Product: JDT
Component: APT (show other bugs)
Version: 4.10
Hardware: PC Windows 10

Importance: P3 critical with 3 votes (vote)
Target Milestone: 4.15
Assignee: Jay Arthanareeswaran ✓ECA
QA Contact:

URL:
Whiteboard:
Keywords:

Depends on: ~~534979~~
Blocks:
Show dependency tree

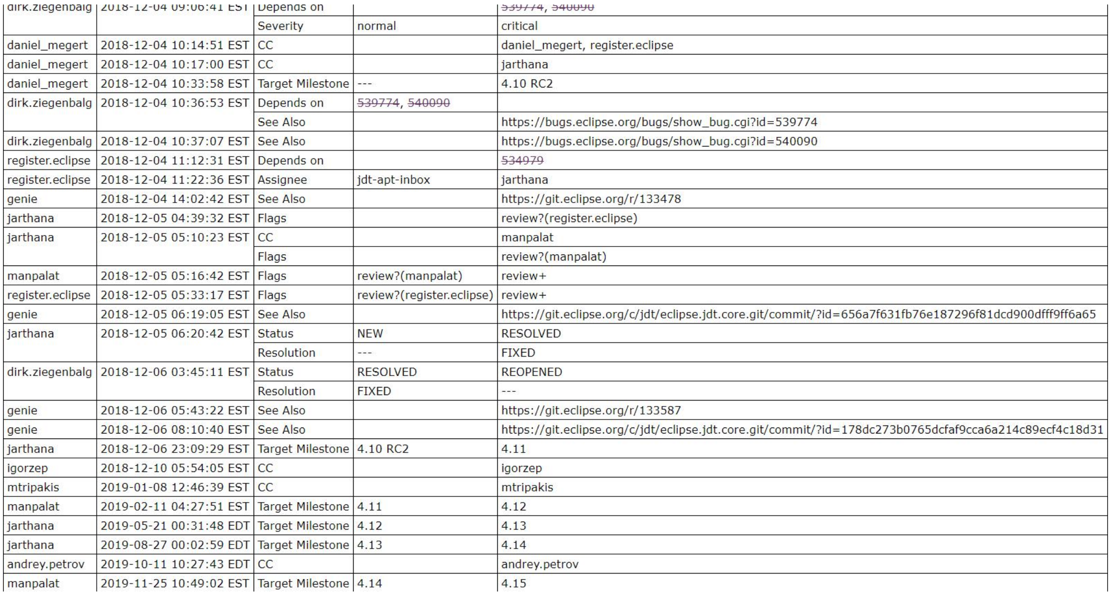
Reported: 2018-12-04 09:02 EST by Dirk Ziegenbalg ✓ECA
Modified: 2019-11-25 10:49 EST (History)
CC List: 7 users (show)

See Also: ~~539774~~
~~540090~~
Gerrit Change
Git Commit
Gerrit Change
Git Commit

Flags: register.eclipse: review+
manpalat: review+

图 5-2 542090 号缺陷报告

从 542090 号报告的提交日期和内容分析，在上述的 bug 被验证修复之后仍然有人发现了相关的 bug 和变化。因此这个新的 bug 变化可以被纳入需求变更的时间线中，而且这个新的 bug 导致了新的修复和更改，可以发现该 bug 也进行了两次新的 commit，按照同样的思路，我们也打开了’Modified“字段查看了变化的历史。



dirk.ziegenbalg	2018-12-04 09:06:41 EST	Depends on		~~539774~~, ~~540090~~
		Severity	normal	critical
daniel_megert	2018-12-04 10:14:51 EST	CC		daniel_megert, register.eclipse
daniel_megert	2018-12-04 10:17:00 EST	CC		jarthana
daniel_megert	2018-12-04 10:33:58 EST	Target Milestone	---	4.10 RC2
dirk.ziegenbalg	2018-12-04 10:36:53 EST	Depends on	~~539774~~, ~~540090~~	
		See Also		https://bugs.eclipse.org/bugs/show_bug.cgi?id=539774
dirk.ziegenbalg	2018-12-04 10:37:07 EST	See Also		https://bugs.eclipse.org/bugs/show_bug.cgi?id=540090
register.eclipse	2018-12-04 11:12:31 EST	Depends on		~~534979~~
register.eclipse	2018-12-04 11:22:36 EST	Assignee	jdt-apt-inbox	jarthana
genie	2018-12-04 14:02:42 EST	See Also		https://git.eclipse.org/r/133478
jarthana	2018-12-05 04:39:32 EST	Flags		review?(register.eclipse)
jarthana	2018-12-05 05:10:23 EST	CC		manpalat
		Flags		review?(manpalat)
manpalat	2018-12-05 05:16:42 EST	Flags	review?(manpalat)	review+
register.eclipse	2018-12-05 05:33:17 EST	Flags	review?(register.eclipse)	review+
genie	2018-12-05 06:19:05 EST	See Also		https://git.eclipse.org/c/jdt/eclipse.jdt.core.git/commit/?id=656a7f631fb76e187296f81dcd900dfff9ff6a65
jarthana	2018-12-05 06:20:42 EST	Status	NEW	RESOLVED
		Resolution	---	FIXED
dirk.ziegenbalg	2018-12-06 03:45:11 EST	Status	RESOLVED	REOPENED
		Resolution	FIXED	---
genie	2018-12-06 05:43:22 EST	See Also		https://git.eclipse.org/r/133587
genie	2018-12-06 08:10:40 EST	See Also		https://git.eclipse.org/c/jdt/eclipse.jdt.core.git/commit/?id=178dc273b0765dcfaf9cca6a214c89ecf4c18d31
jarthana	2018-12-06 23:09:29 EST	Target Milestone	4.10 RC2	4.11
igorzep	2018-12-10 05:54:05 EST	CC		igorzep
mtripakis	2019-01-08 12:46:39 EST	CC		mtripakis
manpalat	2019-02-11 04:27:51 EST	Target Milestone	4.11	4.12
jarthana	2019-05-21 00:31:48 EDT	Target Milestone	4.12	4.13
jarthana	2019-08-27 00:02:59 EDT	Target Milestone	4.13	4.14
andrey.petrov	2019-10-11 10:27:43 EDT	CC		andrey.petrov
manpalat	2019-11-25 10:49:02 EST	Target Milestone	4.14	4.15

图 5-3 542090 号需求修改历史记录

从该报告的历史事件中照样可以发现了”REOPENED”的关键字，说明当时重新开启了该 bug 的修复和变更，不过似乎到目前为止并未完全修复和验证完毕。总之新的更新代码文本和相关内容都会被加入到上文的变更内容中。沿着这个追踪的思路，我们整理了一条完整的变更时间线，该时间线上包括了发现 bug，修复 bug，bug 出现新的报告和变化的过程

2018-10-12 16:35:04 EDT 540090 号 bug 被发现

2018 年 10 月 13 日 10:55:59 EDT 540090 号 bug 通过了第一次 commit 修复

2018-10-13 11:12:03 EDT 540090 号 bug 认为已经修复并更改状态为”RESOLVED”

2018-10-28 10:15:59 EDT 540090 号 bug 出现新的情况并“REOPENED”

2018-11-01 06:29:22 EDT 540090 号 bug 通过了第二次 commit 修复

2018-11-01 06:30:18 EDT 540090 号 bug 认为已经修复并更改状态为”RESOLVED”

2018-12-04 09:06:41 EST 出现了新的变化，有人提出了 542090 号有相关的状况和变化

2018-12-05 06:19:05 EST 542090 号 bug 通过了第一次 commit 修复

2018-12-05 06:20:42 EST 542090 号 bug 认为已经修复并更改状态为”RESOLVED”

2018-10-05 17:27:21 EDT 相关的依赖项 539974 号 bug 被修复

2018-12-06 03:45:11 EST 542090 号 bug 出现新的情况并“REOPENED”

2018-12-06 08:10:40 EST　　542090 号 bug 通过了第二次 commit 修复

2018-12-06 ~ 2019-11-25 10:49:02 EST　　　未完全确认被修复正确，在不断迭代检查

5. 总结

从上文的 bug 的提出-修复-重新开启的过程中可以看到，即使是这样一个简单的异常提示'源文件已创建'的小 bug，也会有反反复复的变化以及修复过程。缺陷的修复也是会出现不断的变更过程，一个缺陷在修复之后还有可能在后续更新后再次出现新的情况，一个缺陷的修复并非一劳永逸而是会不断地发展。建立良好的需求跟踪分析过程，有利于软件开发和后续的维护。

注：小组成员及成绩比例

学号	姓名	成绩比例
171860657	徐浩	0.25
171860677	吴鸿祐	0.25
171868570	周吴成	0.25
171870642	王子豪	0.25