Software Assurance Tips

A product of the Software Assurance Tips Team[1]

Jonathan Hickerson

Monday 19th July, 2021

1 File-by-File Scanning for Ada

Updated Tuesday 27th January, 2026

Scanning code which will not fully build is often a thorn in the side of the software assurance analyst who has been delivered code from an external customer. This is doubly true in a language such as Ada with few viable scanning options.

AdaCore CodePeer has an operation called "Analyze File by File" which sounds like it would help if the entire Ada project will not build. After all, if a portion of the code will build, surely obtaining scan results for that portion of the code is better than no results at all. An enterprising software assurance analyst, full of hope and cheer, might try this tool, only to come to the crushing realization that even though it operates on individual files, it is only intended for use in the context of an already buildable project, sending them back to square one.

A workaround to this issue, and a way to have CodePeer successfully operate on individual files in an Ada project is a little more involved.

While a normal `codepeer-gprbuild` scan operates on a project and expects the full project to build, one can pass it a project and a specific file, and while the project will fail to build, if the individual file builds, it will generate a `.SCIL` file, which is the output of the building/linking process which CodePeer requires for analysis. The command line argument for this setup is shown in Listing 1.

```
$ codepeer-gprbuild -ws -c -f -u -P[a project file] --subdirs=codepeer-file --codepeer
    -gnateC [a .ada, .adb, or .ads file]
```

Listing 1: CodePeer GPRBuild Setup

Once this file has been processed and its `.SCIL` file created, one can do a normal `codepeer-gprbuild` on the entire project from within GPS. While the build will fail, it will still check for existing `.SCIL` files and then perform an analysis on these files. By iteratively performing the above command on all `.ada`, `.ads`, and `.adb` files in the project, and following it with a normal analysis, an analyst can get some results out of CodePeer. The full process is:

1. Build a new Ada project file (a `.gpr` file) which does nothing but include all of the files within the Ada project. An example project is shown in Listing 2.

   ```
   Project Test is
           For languages use ("Ada");
           For Source_Dirs use ("./**");
   End Test;
   ```

   Listing 2: All-Inclusive GPR Example

2. Run the command in Listing 1 against each file that requires CodePeer analysis. This is probably best done with a script, such as one demonstrated in Listings 3 and 4 where `Project.gpr` is the project created from Listing 2.

   ```
   #!/bin/bash
   find ./ -name "*.ad[abs]" | while read x; do
           echo ${x}
           codepeer-gprbuild -ws -c -f -u -PProject.gpr --subdirs=codepeer-file --
               codepeer -gnateC "${x}"
   done
   ```

   Listing 3: Bash Script for Ada Single-Build Automation

   ```
   @ECHO OFF
   for /R %%f in (.\*) do (
           echo %%~xf|findstr /r "ad[abs]$" > nul 2>&1
           if errorlevel 0 (
                   echo "%%~ff"
                   codepeer-gprbuild -ws -c -f -u -PProject.gpr --subdirs=codepeer-
                       file --codepeer -gnateC "%%~ff"
   ```

```
                    )
        )
```

Listing 4: Batch Script for Ada Single-Build Automation

3. In the GPS studio, open the project file, and select *analyze*. Step one of the analysis process (building and linking) will fail, but when step two begins, it will detect the `.SCIL` files from above and proceed to analyze those files.

4. In the GPS studio, export the analysis results.

I hope this makes the reader’s next adventure in the world of unbuildable Ada code a little less painful!

References

[1] Jon Hood, ed. SwATips. `https://www.SwATips.com/`.