GOES-R AIT Algorithm Acceptance Procedure

The purpose of this document is to list and explain the steps to follow when an AWG algorithm is received by the AIT. These steps will ensure a consistent transition from the AWG development environments to the AIT development environment.

Algorithm Acceptance Procedure

1. Check-out from CM
   Check-out a working copy of the algorithm package from revision control.

2. Makefile
   Verify a Makefile exists to build the entire package. Modify the Makefile to use our compilers.

3. Compile
   Compile the source code with our compilers.
   - Linux
     - Compile with PGI
     - Compile with Intel
   - IBM
     - Compile with XL

4. Verify Output
   Run the code as-is to verify our output matches their output.

5. Valgrind
   Valgrind is a suite of tools for debugging and profiling Linux programs. Memcheck is one of these tools which can check for memory errors. Compile your program with -g to include debugging information so that Memcheck's error messages include exact line numbers.

   If you normally run your program like this:

   ```
   myprog arg1 arg2
   ```

   Use this command line:

   ```
   valgrind --leak-check=yes myprog arg1 arg2
   ```

   Memcheck is the default tool. The --leak-check option turns on the detailed memory leak detector.

   Your program will run much slower (eg. 20 to 30 times) than normal, and use a lot more memory. Memcheck will issue messages about memory errors and leaks that it detects

6. Profiling
   Profiling gives an indication of how much time is spent in each function. Compile the source code with the –pg option (for gprof use) or the –Mprof=func option (for pgprof use). Run the code in its normal way and when it is done there should be a gmon.out (or pgprof.out if –Mprof=func was used) file to analyze with gprof (or pgprof if –Mprof=func was used).

7. Forcheck (fortran code)
   Use Forecheck on Fortran code to check for non-standard source code and other bugs that may not have been caught by the compiler.

8. Lint (C code)
   Use lint (or splint, which claims to be a better lint) on C code to check for common programming mistakes.

9. Code Checker
   Use our in house code checking tools to verify the source code meets our coding standards.

10. Commentary
    Perform a Peer Review to make sure comments in the code are readable, concise, and informative.

11. Check-in to CM
    Check-in the updated algorithm package to revision control.