# Workflow Engine Evaluation

**Product**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**URL:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Evaluator:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Date:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Instructions: Fill out the above information. For each item below, give a score from 1 (does not meet requirement) to 10 (fully meets requirement) in the blank right after the bullet. Then provide supporting info in the space following the item’s text, if possible.*

1. **\_\_\_\_\_ Open source.** The product must be free to use, and should offer source code access. A viral GPL license would be a problem; prefer an Apache or BSD license that does not limit our options. Please state license in the space below.
2. **\_\_\_\_\_ Supported product**. Should be an active, thriving project. Should have corporate sponsorship as evidence of lasting support.
3. **\_\_\_\_\_ Portable**. Must run on Windows, Mac, Linux, and NGP (which is just modified Linux, one imagines.)
4. **\_\_\_\_\_ Open standards.** Usage of documented open standards for workflow storage, communication protocols, query language, graphics, etc is a plus. Proprietary/closed standards are a big negative, especially binary file formats.
5. **\_\_\_\_\_ Java-based.** Engine itself must be Java-based, or Java friendly, as we’ll want to embed it in Java, and extend it with Java.
6. **\_\_\_\_\_ Python-friendly**. At a minimum it should be easy to invoke workflow actions consisting of Python scripts.
7. **\_\_\_\_\_ Graphical builder.** The product should have a full-featured graphical workflow builder. A big plus if builder is SWT-compatible so it can be integrated in SAW.
8. \_\_\_\_\_ **Data tools.** The graphical builder should include tools for extracting data from files as part of workflow, like iSight
9. **\_\_\_\_\_ Easy to implement workflow actions**. The engine must be able to execute external programs as actions. It should be able to use “plugins” written in Java. “Plugins” written in Python would be nice to have. It must have well-documented API or mechanism for defining actions.
10. **\_\_\_\_\_ Abstract data channels.** Must support the notion of an abstract data channel, explicitly or implicitly. The number one consequence of this is that exchanged data mustn’t need to pass through engine or through a file. The builder should support explicit “type” for data links between workflow actions.
11. **\_\_\_\_\_ Workflow persistence**. Workflows should be persistent during execution to facilitate restart, progress feedback, and reporting. Bonus points if there are existing reporting tools that support getting workflow status on your iPhone.
12. **\_\_\_\_\_ Workflow storage**. Builder must save workflow definition in a portable file, and engine must read entire workflow definition from this file. Files should have an open/standard format. We should be able to save workflow action definitions in a “library” for reuse.
13. **\_\_\_\_\_ Scalability. Engine and builder** must handle hundreds or thousands of nodes in a workflow. Engine must support running multiple workflows simultaneously. Engine must work with exascale problems.
14. **\_\_\_\_\_ Headless mode.** Workflow execution must support headless (no-GUI) mode on all platforms
15. **\_\_\_\_\_ Composable workflows.** Engine and builder should have support for nested or composable workflows
16. **\_\_\_\_\_ Distributed operation.** Engine should support distributed operation. Just executing actions via ssh would be fine, as long as we can leverage Kerberos authentication.