Auto Pilot Requirements Specification

1. Scope

1.1 Overview

Auto Pilot makes use of the API defined by Phase Management to automate the project execution. Scheduled phases will be started if certain conditions are met, and open phases will be ended if certain conditions are met. Phase execution will be evaluated periodically or on events. Phase changes will be audited.

1.2 Logic Requirements

1.2.1 Auto Pilot Switch

Auto pilot will only work for the active projects that have the auto pilot switch on in extended properties. Phases for project without this option will need to be executed explicitly.

1.2.2 Polling Interval

Auto pilot will poll the projects at a configurable interval. The default value will be 5 minutes. It should run at every interval starting from midnight. Designer is encouraged to make use of the TopCoder Job Scheduler to achieve this purpose.

1.2.3 Event Mode

Auto pilot can also be fired programmatically. Component user can invoke auto pilot on a per project basis.

1.2.4 Advance Project

If any of the project's open phases can be ended, auto pilot should end the open phase. Likewise, if any of the project's scheduled phases can be started, auto pilot should start the scheduled phase. Phase changes must be applied until no change can be changed.

1.2.5 Command Line

The component should provide a command line interface.

1.2.6 Auditing

Each time a phase is started or ended an audit entry should be made, which includes the project, phase, operation and a timestamp.

1.3 Required Algorithms

Algorithm to advance project should be described. The algorithm should minimize the number of phase operation queries in order to move the phase to the destination phases.

1.4 Example of the Software Usage

A project can go from alpha phase into beta when all the major bugs have been fixed. Auto pilot is configured to run once each day in this scenario. After the phase handler for the alpha phase confirms that all major bugs are resolved, and auto pilot closes the alpha phase. A second phase handler indicates that beta has all its dependencies closed and auto pilot opens the beta phase. Upon opening the phase the phase handler sends notification emails to all project resources.

1.5 Future Component Direction

More sophisticated triggering mechanism could be devised.

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

None.

2.1.2 External Interfaces

None.

2.1.3 Environment Requirements

- Development language: Java1.4
- Compile target: Java1.4

2.1.4 Package Structure

com.topcoder.management.phase.autopilot

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

None.

3.2 Technical Constraints

3.2.1 Are there particular frameworks or standards that are required?

None.

3.2.2 TopCoder Software Component Dependencies:

- Project Phases
- Project Management
- Phase Management
- Deliverable Management
- Job Scheduler

**Please review the TopCoder Software component catalog for existing components that can be used in the design.

3.2.3 Third Party Component, Library, or Product Dependencies:

None.

3.2.4 QA Environment:

- Solaris 7
- RedHat Linux 7.1
- Windows 2000
- Windows 2003

3.3 Design Constraints

The component design and development solutions must adhere to the guidelines as outlined in the TopCoder Software Component Guidelines.

3.4 Required Documentation

3.4.1 Design Documentation

- Use-Case Diagram
- Class Diagram
- Sequence Diagram
- Component Specification

3.4.2 Help / User Documentation

- Design documents must clearly define intended component usage in the ‘Documentation’ tab of Poseidon.