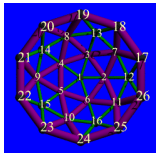

Buckminster | Component Assembly Project

Eclipse Technology Project Proposal
Project Creation Review
May 23, 2005

buckminster summary

- Addresses development problems associated with assembling complex component structures in team-based development
- Proposed for incubation as an Eclipse Technology Project Sub-Project
- Sponsoring entity: ObjectWeb consortium
- Project name: after Buckminster Fuller, architect, engineer; inventor of the geodesic dome; pioneer of manufactured modular structures.

key problem: working with complex component structures

- Development projects increasingly involve component structures:
  - with complex, version-specific dependencies and relationships
  - combining components from open source and proprietary projects
  - characterized by a high rate of change
  - mixing code artifacts in a range of forms and formats
  - resourced with virtual development teams and heterogenous development infrastructure
- Eclipse currently provides an excellent environment for managing component complexity in plug-in development, but:
  - PDE management support is available only for plug-ins and other Eclipse component types
  - Project-Sets lack support for version dependency scenarios
  - Project Sets lack support for flexible location resolution

key objectives

- Buckminster's primary objective is to leverage & extend Eclipse to:
  - bring complex component development on par with current mechanisms for plug-in & feature development
  - extend the component dependency model to allow materialization based on match rules
- Buckminster will accomplish this by:
  - introducing a project/component-agnostic way of describing arbitrarily complex component structures and dependencies in development projects
  - allowing component materialization based on match rules, i.e.similar to platform mechanism for runtime resolution of plug-ins/features
  - providing a materialization mechanism that handles all component types referenced through repository handlers

scope & implementation

- Buckminster's key features will include:
  - Complex dependency resolution
  - Uniform component dependency formats
  - Intelligent retrieval mechanisms
  - Flexible project workspace binding
- Buckminster will be implemented as an Eclipse feature/plug-in set in order to:
  - work seamlessly inside the Eclipse environment &
  - be packageable as a 'product' for pure command line access
- Extension points will be supplied for community participation

drill-down: complex dependency resolution

- Provides recursive resolution of dependencies
- Supports a variety of versioning schemes
- Applicable to source and binary artifacts that are not version-controlled in a traditional sense
- Uses match rules similar to those in the Eclipse plug-in runtime framework, allowing comparison of current and prior dependency resolutions to support update impact analyses

drill-down: uniform component dependency format

- Component-type agnostic mechanism for describing components and their respective targets and dependency requirements
- Will leverage dependency info associated with typical Eclipse projects and range of other component types
- Extensible to provide additional strategies for dependency pattern recognition

drill-down: intelligent retrieval mechanisms

- Leverages the Eclipse "Team Project Set" mechanism by separating bill of material needed for given configuration from its actual materialization
- Separation is of value since:
  - dependencies may appoint software that is locally installed on one machine but lacking on another
  - bills of materials may be shared between team members, while materialization info may vary
  - information about repositories will be abstracted out in order to provide site and repository transparency

drill-down: flexible workspace binding

- Allows a component materialized on disc to be bound to a workspace in different ways
- Supports "Proxy Projects" consisting of:
  - links to physical artifacts +
  - auto-generated Eclipse project information
- Approach is of value when sharing code or other artifacts that are not Eclipse projects

project roadmap

- Preliminary scope of currently planned milestones is as follows:
  - Milestone 1 (Sep 05)
    - Component resolution and materialization using commonly used repositories, both source code control systems and URL based repositories
    - Recognition of standard Eclipse PDE-components
    - Support for basic version resolution strategies
    - Implemented as wizards (Eclipse UI) and command line (Eclipse UI & Product)
    - Transparent integration with Eclipse build system
  - Milestone 2 (Jan 06)
    - Recognition of J2EE, Maven and other common component types
    - Support for advanced version resolution strategies
    - Graphical editor-set for range of supported configuration specification formats
  - Milestone 3 (May 06)
    - Support for wider range of commercial repository providers and component types
- Final milestones will reflect input from community; project will proactively seek input from other Eclipse projects.

Team

- Buckminster proposal sponsored by ObjectWeb consortium
- Initial project team includes the following individuals:
  - Thomas Hallgren
  - Kenneth Ölwing
  - Pontus Rydin
  - Mitch Sonies
- Additional individuals have expressed interest in becoming committers

project transition

- Assuming success as a Technology sub-project, either of the following scenarios is plausible long term:
  - Buckminster is merged into the mainline Eclipse Platform top-level project
  - Buckminster is moved to a permanent state as a Tools Platform project

Thank You

The Buckminster Team