



The p2 Provisioning Platform


Eclipse European Summit - Ludwigsburg

Oct 27-29, 2009


Henrik Lindberg, Cloudsmith Inc



Provision what – where



Engineers

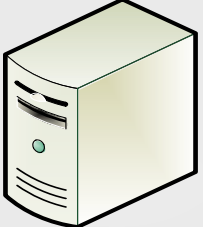


Users


source code
open source
in-house built
licensed software



applications
tools
runtimes



Automated Builds
Test



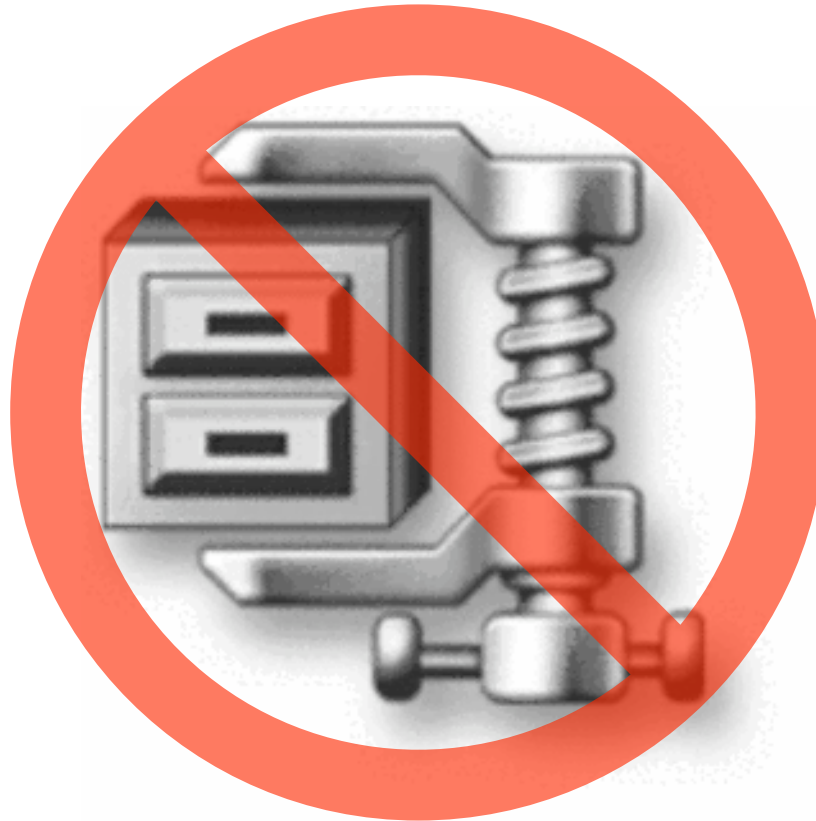
Test



Wide range of requirements

- packaged applications – source code
- strict policy control – full flexibility
- approved repositories – any repository
- fixed configuration – dynamic configuration
- fully automatic – user driven – (headless)
- update to latest – update to policy controlled (downgrade)

Installation state of the art < Eclipse 3.4



Unzipping is not installing!



Eclipse and Provisioning

p2



p2 is installing!

- A replacement for the old Update Manager
 - New UI, simplified workflow
 - Manage Eclipse, RCP and more
 - exe, ini, bundles, registry keys, settings, native code, ...
 - Shared bundles across Eclipse-based products
 - An installer
- A provisioning solution for OSGi™ systems
 - Managing non-running instance
 - Start level, framework extension
 - Fine-grained dependency management
- Extensible
- GUI and Headless
- One consistent model



New user interactions

- Simple update workflow
 - Replace multi-steps wizards
- New metaphors
 - Drag n Drop for install, adding repos, ...
- More flexible repositories
 - Connect to p2 repos, Update Sites, (OBR, Maven, ...)
- Managed folders
 - Explicit “watched” locations
 - Drop content to have it installed
 - No need to unzip and -clean



From Install to Update

- Installing and Updating are the same operations
- Installed shape same as zip shape
 - Flexibility in delivery
- Programmatic API for all operations

**p2 offers a continuum
from installation to updating**



Concepts and Architecture

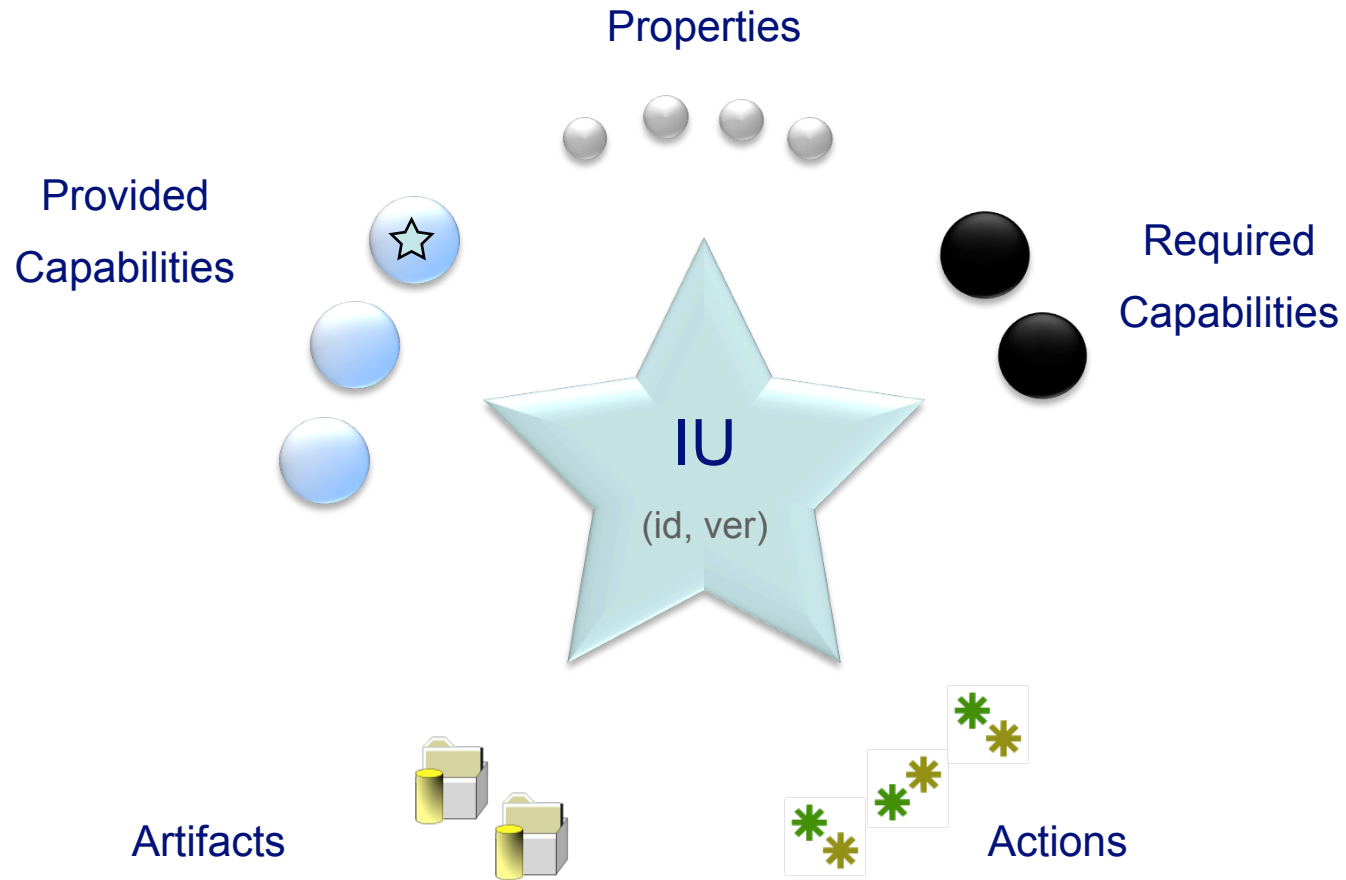
Installable Units

Decouple decision making from the actual content

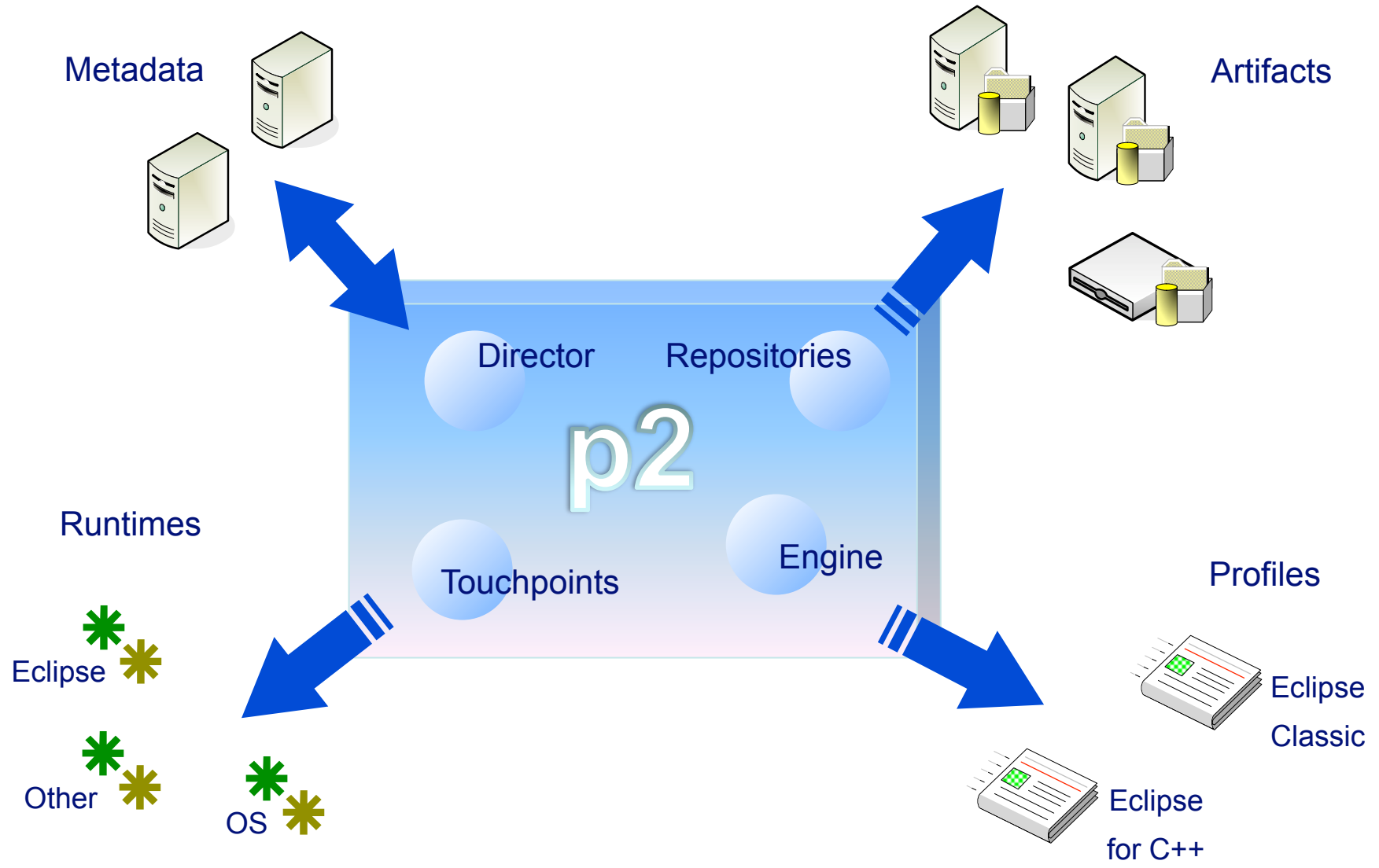


Everything is an IU
Everything is installable

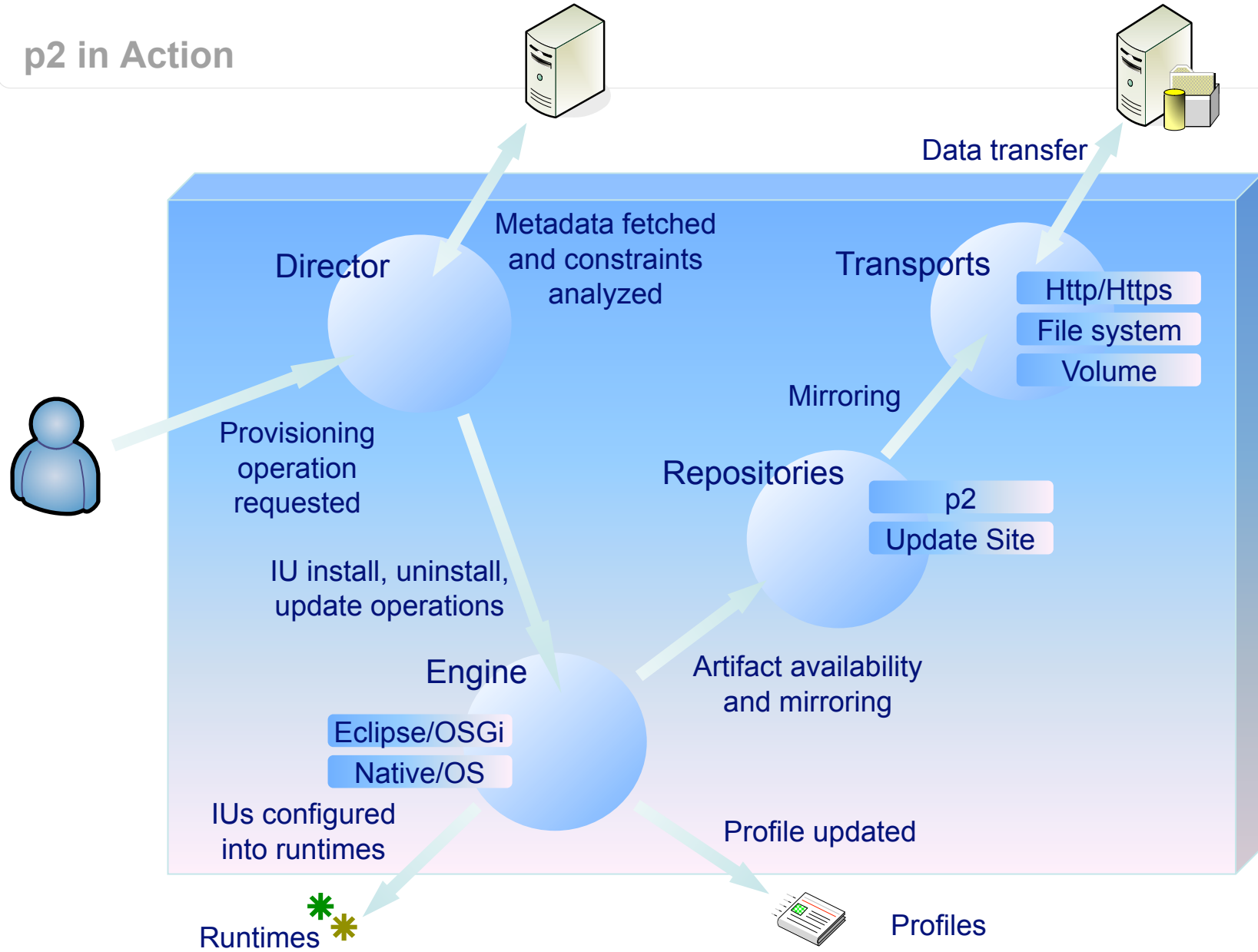
Anatomy of an IU



p2 Architecture



p2 in Action



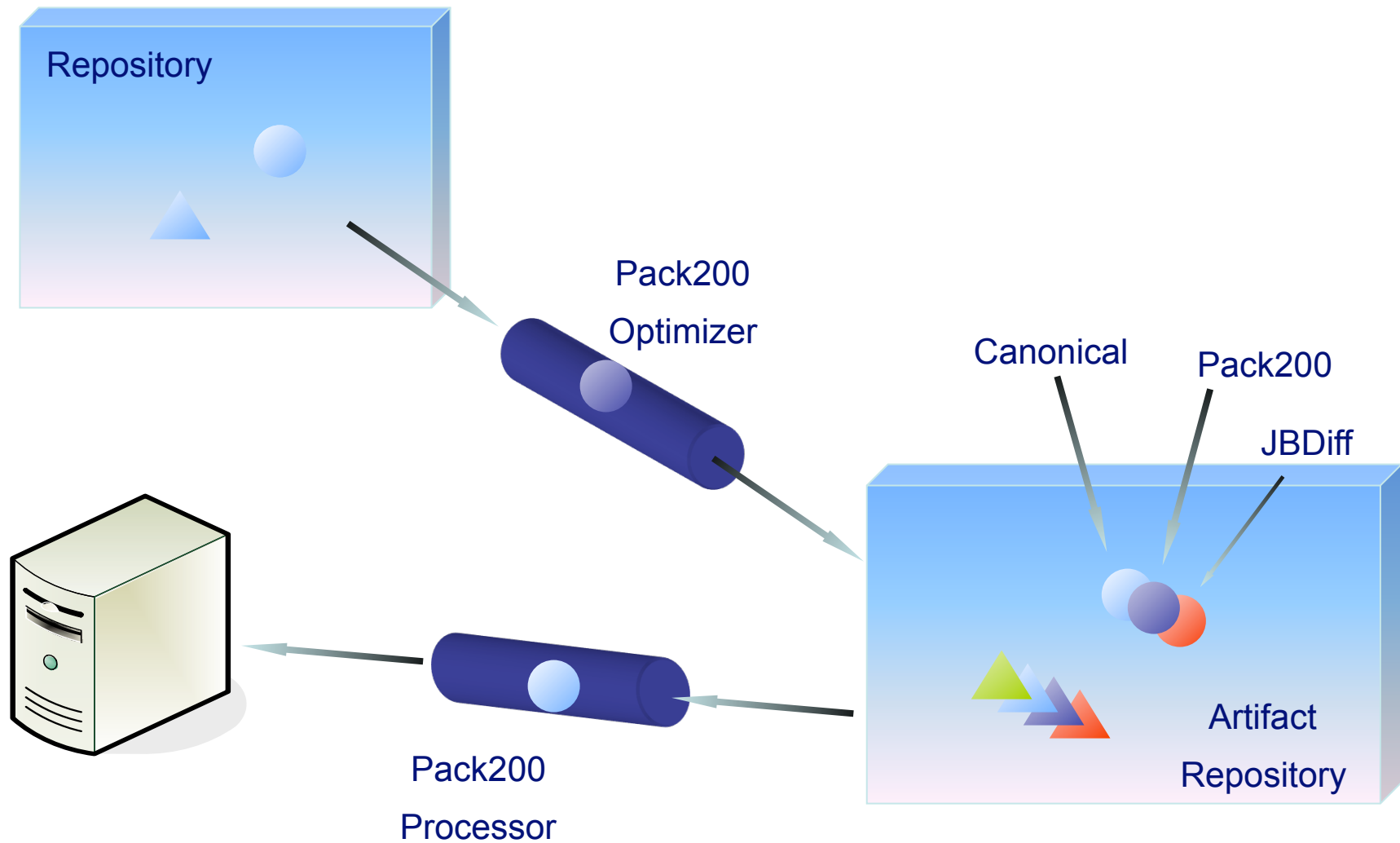
SAT-based resolution

- The installation problem is NP-complete
 - Map dependencies to Boolean formula
 - Use SAT4J
 - If there is no solution it will tell us
 - If there is a solution we will get an optimal one



Special thanks to
Daniel LeBerre
and Anne Parrain

Optimizing and Processing Artifacts





p2 releases



Releases

- **Eclipse 3.4 - Ganymede**
 - Replace Update Manager
 - Improved functionality
 - Minimal disruption
 - Starting point for new provisioning platform
 - Provisional API
 - “Client side” only
- **Eclipse 3.5 - Galileo**
 - Themes: Robustness, completeness, performance, ease of use
 - Flexible options for inclusion in RCP applications
 - Composite repositories
 - Omni Version
- **Eclipse 3.6 - Helios**
 - API, Java 1.5
 - Internal cleanup
 - Ease of use in API
 - Availability on non Equinox OSGi implementations



Summary

- p2 is a powerful provisioning platform
 - Highly extensible
 - Enable the creation of comprehensive solutions
 - The basis of Eclipse provisioning going forward
-
- <mailto:p2-dev@eclipse.org>
 - <http://wiki.eclipse.org/Equinox/p2>



Terminology

- p2 / Agent
 - The provisioning infrastructure on client machines
- Installable Unit (IU)
 - Metadata that describes things that can be installed/configured
- Artifact
 - The actual content being installed/configured(e.g., bundle JARs)
- Repository
 - A store of metadata or artifacts
- Profile
 - The target of install/management operations
- Director
 - The decision-making entity in the provisioning system
- Engine
 - The mechanism for executing provisioning requests
- Touchpoints
 - Integration with particular runtime or management systems