The Actual Real World at EclipseCon/ALM
Raise your Hand if you are Sure
Addressing the Issues
Real World: Tycho

Issues

- World Wide Distributed Teams
  - India, China, Europe, Argentina, United States
  - Differing Band Widths and Restrictions on sites and content that can be accessed.
- Differing levels of knowledge when it comes to P2, OSGi, Maven, and Tycho
- A Bag of Tricks – Everybody has their own favorite tool they like to use
Real World: Tycho Issues

- Duplicated configurations across the tycho/osgi projects.
- Differing versions of Tycho being used depending on project
  - 0.9.0, 0.10.0, 0.11.0, 0.12.0
- Composite Repositories being over used.
- No Nexus or Artifactory repository in use.
  - Ibiblio used as main external repo.
Release Engineering
Defense Mechanism
Composite Repos

- Useful for Development Snapshots
  - Particularly when using symbolic links.
  - Can be useful in QA repositories depending on if pointing to static repositories and directory locations.

- Prefer Aggregated repositories for deployment and final production.
  - Composites can be used but entries need to remain static.
Real World: Tycho
The Good

- Continuous Integration Tools in Place
  - Git, Jenkins, GitHub
- Automated Processes to deploy repos
  - Mirroring repos to various locations.
  - https://github.com/intalio/tycho-p2-scripts
  - Composite Repo construction, Deployment, Versioning,
  - Release scripts
Embrace Parent POMs

- Consolidate your tycho configurations that are common across the enterprise into a versioned parent-pom file.
- Makes upgrading your projects to use new versions of tycho easier.
- Provides some consistency across the builds for various projects.
- Use maven properties liberally.
  - You can always override in a child pom if necessary.
Deploying a Repository

- Embrace Maven Profiles they are your friend!
- Example Release Profile:
  - [https://gist.github.com/kingargyle/5232227](https://gist.github.com/kingargyle/5232227)
  - Copies the created product or repository to a specified location.
  - Creates symbolic links to the most current repo if multiple are versioned and deployed.
- RSYNC is your friend for mirroring.
- Mirror Jobs should be triggered as part of your deployment process, automate it.
Maven Central and P2

- Want to use Maven Central and Nexus for third party dependencies.
- Not everything has an OSGI manifest.
- Don't want to manually massage items (ala Eclipse Orbit)
- Eclipse Orbit repos don't have everything due to IP also not stable repos (i.e. not there forever)
P2-MAVEN-PLUGIN

- https://github.com/reficio/p2-maven-plugin
- Create P2 repositories from items in maven central.
- Adds necessary OSGi manifest
- Uses Tycho repository plugins
- Uses BND to generate the OSGi manifest
- Highly configurable
- Will create source bundles if source can be found.
Advantages

• Simplified our builds.
  – P2 repos only used throughout the OSGi builds now.

• Eliminated the needed for the option to consider POM dependencies.
  – This is just messy, avoid this if possible!

• Allowed for easy inclusion of new bundles and sources.

• No need to recompile the artifacts and no need for separate builds or p2 repos.
Questions?
Suggestions

- Don't fight Maven!
  - Learn the different build phases
  - Don't re-invent the wheel. There is a plugin probably for it already.
  - Embrace Profiles
- Leverage Maven Central
  - Setup internal Nexus Mirrors
  - Deploy to various Regional Locations.
- Use the p2-maven-plugin it will save you time and effort.
Disadvantage

- Depending on number of dependencies and if artifacts are cached locally or not, can take a while to fetch and build the initial repository.
- Some people still don't like Maven and XML
- p2-maven-plugin currently not in maven central.
  - Build it yourself and deploy to Nexus
  - Mirror the developers p2-repo on github.