DSDP Mobile Tools for Java 0.9 Release

Eclipse Development Process version 2.4 – August 20, 20078
Slide deck v3 – Aug 30, 2008

Release Review Date: Oct. 08, 2008
Communication Channel: DSDP MTJ Newsgroup
(http://www.eclipse.org/newsportal/thread.php?group=eclipse.dsdp.mtj)

Christian Kurzke & Gustavo de Paula

Release Review Agenda

- MTJ Overview and History
- Features
- Non-Code Aspects
- API Status
- Architectural Issues
- Tool Usability
- End-of-life
- Bugzilla
- Standards
- UI Usability
- Schedule
- Communities
- IP Issues
- Next Release Project Plan

A Brief MTJ History

- MTJ Project was created by Nokia in 2005
  - Main project sponsors were Nokia and IBM
- Mobile application development environment
  - **CLDC** and **CDC** based devices
  - **Deploy** and **Execute** an application
- Focus was on providing a **framework** that other vendors can extend to create their own JavaME tools
- Release 1.0 was initially planned to **September 2007**
- MTJ current release is 0.7 from **November 2006**

- Currently renewed discussions about focus of MTJ, new potential contributors, Motorola, RIM
  - MTJ Reboot
  - Evaluate other option to MTJ, such as **EclipseME**

EclipseME – Current Option in Eclipse World

- EclipseME is an Eclipse plug-in for JavaME development
  - Focus on **CLDC/MIDP** (other profiles could be supported)
  - Provides all **basic services** (build, sign, obfuscate, etc.)
  - Provide **some extensibility**, but it is not its main focus
- First public releases on **2003**
- Current version is **1.7.8**
  - Almost **500.000** downloads
- Created and maintained by **Craig Setera**

- Used by all **major** mobile device **manufactures**
  - Nokia, Motorola, Sony Ericsson, etc.

- EclipseME focus is the opposite of original MTJ focus
  - Framework vs. Tool

MTJ Reboot Plan

- Use **EclipseME 1.7.8** as a base for the MTJ development
  - **Less risk**
  - **Large** user community
  - Easier to **create a developer community**
- Re-structure EclipseME code to make it more **flexible**
  - Deployment
  - Signing
  - Packing
- Future: **Port** all interesting features that are already implemented on current **MTJ 0.7** into MTJ 1.0

Features

- 0.9 project plan with Milestones and detailed Deliverables available at http://www.eclipse.org/projects/project-plan.php?projectid=dsdp.mtj
- Main focus is to provide at least the same features that were available on EclipseME
  - Support UEI SDKs
  - Support JavaSE SDKs (Miocroemu and Mpowerplayer)
  - Multiple host support
  - Preprocessing of Java code
  - Build, run & debug MIDlet Suites
- Bug Fixing
- Improve developer workflow when compared to EclipseME
  - Enhanced Java Application Descriptor (JAD) Editor
  - Import EclipseME projects
  - Import Netbeans projects
  - Automatic Javadoc discovery on SDK import
  - MTJ Perspective

Non-Code Aspects

- User documentation and tutorials
  - http://dsdp.eclipse.org/help/latest/topic//org.eclipse.mtj.doc.user/html/gettingstarted/gettingstarted.html
  - Automatically updated from nightly builds
- Requirements and system test cases
  - Requirements available at http://wiki.eclipse.org/DSDP/MTJ/Requirements
  - Manual System test cases available at http://wiki.eclipse.org/DSDP/MTJ/Tests
  - All documents reviewed with the community
- ISV documentation
  - Planned to have a Webinar
  - Build Notes with each Milestone
- Working Example Code
  - Add a SDK, Add a Jad editor Page
- Conference talks

API Status

- Initial focus on MTJ is not to provide extensibility
  - This is the objective of the next release
- Currently there are two APIs on MTJ
  - Device Importer/Editor: original from EclipseME
  - Extend JAD Editor: Added on MTJ 0.9

- Device Importer/Editor
  - The API is the same from EclipseME
  - MTJ Provide 3 different implementation of this API
    - One for each type of SDK that is currently Supported
  - Currently there are no unit tests
  - This API might be re-factored on MTJ 1.0
- JAD Editor Extension
  - New API that was added on MTJ
  - Extend current JAD Editor to add new JAD Attributes
  - Currently there are no unit tests

Architectural Issues

- EclipseME code was not refactored
  - Need to split the code into UI and core layers
  - Create an API that can be easily extended
- MIDlet Suites Build System
  - Current build system is complex and hard to maintain and extend
  - When the code is re-factored the build system is one of its main parts
- Need Unit Tests
  - Original EclipseME did not have a lot of unit tests
  - This effort depends on creating a well defined API

Tool Usability

- MTJ inherited all the good usability that was already available on EclipseME
- Developer is able to execute all main operations that are associated with JavaME development
  - Create MIDlet suites and MIDlets
  - Edit JAD file
  - Build & Sign the MIDlet suite
  - Import / Edit JavaME SDKs
  - Run & Debug MIDlet
- Several improvements were made on the workflow to make it easy to execute some of those tasks
  - Create a MTJ perspective:
    - Easier to access create MIDlet suite & MIDlet
  - Add shortcuts to run / debug MIDlets on all the ways supported by the SDKs
  - Enhanced JAD Editor
    - Look&feel closer to other Eclipse projects
    - Central place to execute all tasks

End-of-life

- Since the code base was changed, All MTJ 0.7 APIs are no longer supported,

- org.eclipse.mtj.admin.gui.provider
- org.eclipse.mtj.build.management
- org.eclipse.mtj.build.provider
- org.eclipse.mtj.deployment.management
- org.eclipse.mtj.deployment.provider
- org.eclipse.mtj.device.description.provider
- org.eclipse.mtj.device.description.store
- org.eclipse.mtj.device.management
- org.eclipse.mtj.device.platform.provider
- org.eclipse.mtj.drm.encoding.provider
- org.eclipse.mtj.gui.builder.management
- org.eclipse.mtj.gui.builder.provider
- org.eclipse.mtj.libraries
- org.eclipse.mtj.localization.provider
- org.eclipse.mtj.obfuscation.provider
- org.eclipse.mtj.packaging.provider
- org.eclipse.mtj.persistent.store.provider
- org.eclipse.mtj.preprocessing.provider
- org.eclipse.mtj.preverification.provider
- org.eclipse.mtj.screen.engine.provider
- org.eclipse.mtj.security.management
- org.eclipse.mtj.signing.provider
- org.eclipse.mtj.ui.project.extension

Bugzilla

- 77 bugs resolved up to 10-SEP-2008
- 3 main bugs categories
  - EclipseME Equivalence → port features that are already available on EclipseME
  - Eclipse Infra-structure → adapt project to all Eclipse infra-structure (build system, documentation, etc.)
  - Fix and Improve → fix issues that were already on EclipseME original code, do some improvements on UI and add new small features
- Extensive and open discussions on bugzilla
- No major bug on final 0.9 release

Standards

- Mobile Information Device Profile Specification
  - Version 2.1: http://www.jcp.org/en/jsr/detail?id=118
- Connected Limited Device Configuration Specification
  - Version 1.1: http://www.jcp.org/en/jsr/detail?id=139
- UEI Specification is implemented on MTJ
  - Version 1.0.2 http://java.sun.com/j2me/docs/uei_specs.pdf

UI Usability

- Externalization and Accessibility were not the focus of this release
- This is on the scope of MTJ 1.0

Schedule

- Original project plan posted on 21-Jul-2008
- Revised with MTJ community in the following weeks
  - Full scope of original plan was accomplished
  - Besides that some out of scope features were included (such as code pre-processing)
  - Other new features were also able to be implemented
- All milestone dates were hit, except for the first one
  - Issues with our build system
- Focus on high priority issues. All low priority and re-factoring issues postponed to release 1.0

Communities

- Contributors
  - Initial code base from EclipseME
  - Current Major code contribution from Motorola and Sybase
  - 6 active committers: 3 from Motorola, 2 from Sybase and 1 individual
  - Mailing list participation from Motorola, Sybase, RIM and other individuals
- Adopters
  - This is the first MTJ release
  - EclipseME was already adopted by all major device manufactures
  - MTJ 0.9 is planned to be adopted by MOTODEV Studio for JavaME v2.0
- Users
  - Approximately 4000 downloads of all nightly builds
  - Press release (still being written)

IP Issues

As per the Eclipse IP Policy, the project verifies that:

- ... the “about” files and use licenses are in place as per the Guidelines
- ... all contributions (code, documentation, images, etc) have been committed by individuals who are Members of the Foundation and are abiding by the Eclipse IP Policy (training through Committer HOWTO)
- ... third-party libraries, have been documented in the release and reviewed by the Foundation's legal staff
- ... all contribution questionnaires have been completed
- ... the "provider" field of each plug-in is set to “Eclipse.org"
- ... the "copyright" of each plug-in is set to the copyright owner
- …there are no 3rd party logos or fonts to be licensed under the EPL
- See the IP Log at
  http://www.eclipse.org/dsdp/mtj/development/mtj-log.csv
  http://www.eclipse.org/dsdp/mtj/ipLog.php

Next Release Project Plan

- Join Eclipse Galileo Train
- Project Plan is already on CVS
  - http://dev.eclipse.org/viewcvs/index.cgi/www/dsdp/mtj/development/mtj_plan_1.0.xml

Thank You