

*Creating an End-to-End Mobile Linux
Development and Simulation
Environment Using Eclipse/Tml*

Christian Kurzke

Architect, MOTODEV Studio
Motorola, Inc.

Eric Cloninger

Sr. Product Manager, MOTODEV Studio
Motorola, Inc.



Outline

The Importance of Eclipse and TmL

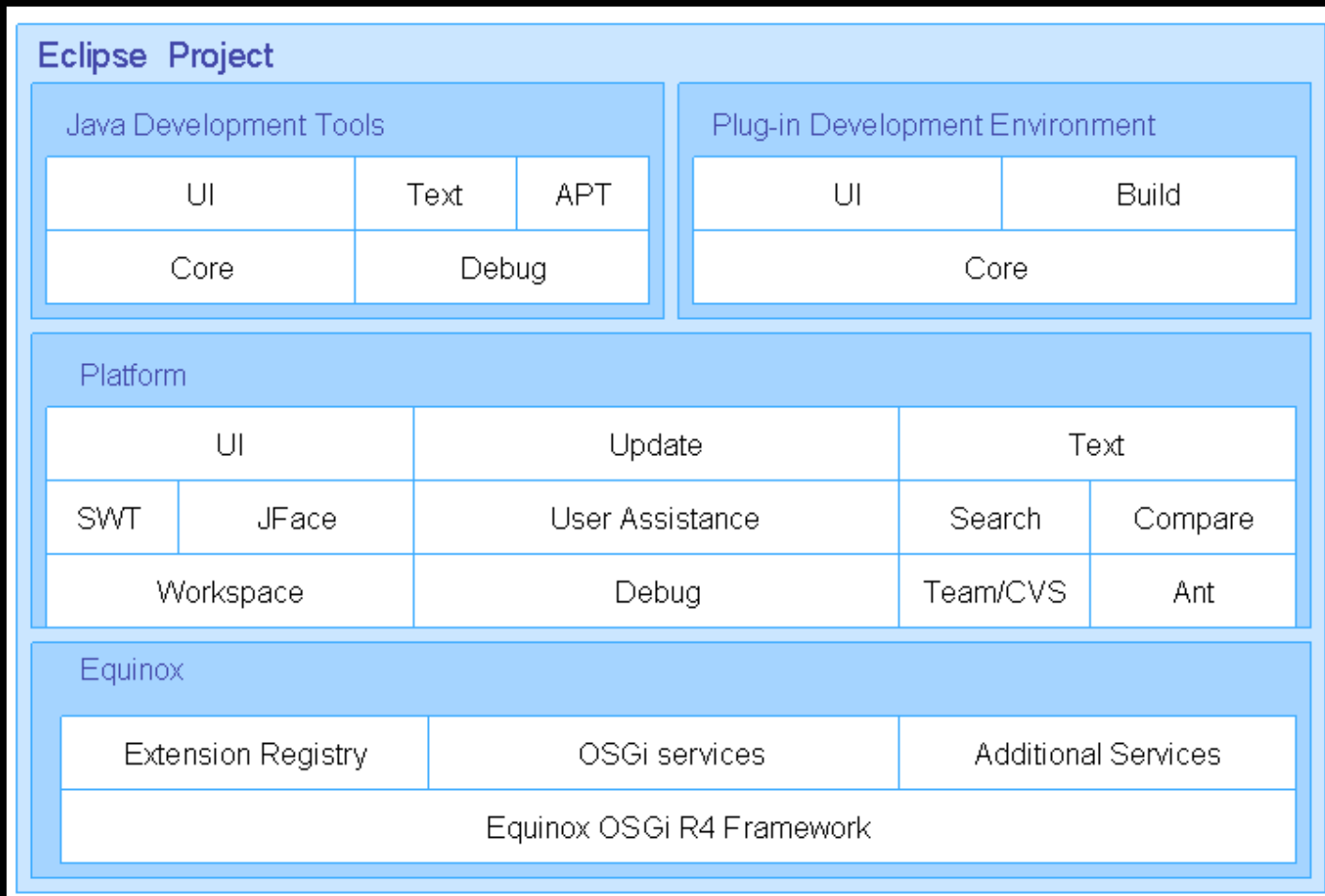
Nature of an End-to-End Mobile Linux IDE

Overview of current Implementation

Future Vision



What is Eclipse anyways



Why Eclipse is Important to Motorola

The goal is to deliver an **integrated** development environment for mobile Linux based devices.

Eclipse provides:

An extensible Base IDE framework

Editors, Code management, plug-in ecosystem

Business friendly License (EPL)

Open Source ecosystem to collaborate on extensions



Why Eclipse is Important to the Linux Developer

Traditional Linux Development done with tools like vi, emacs, cvs, make, gdb, etc.

- Setup and learning curve for each tool

- Debugging (esp. remote debugging) often inefficient and not done (using “printf” instead)

Eclipse simplifies development lifecycle

- unified interface to all tools

- easy to extend with plugins for e.g. CVS, SVN, P4, etc.

- out of the box support for C/C++ competitive with “professional” tools

 - Refactor, code complete, templates, reference checks

 - Syntax highlighting, source block folding, spell check



Why Eclipse is Important to the Mobile Developer

Development for Mobile Devices requires many other non standard tools like:

- Packaging / Signing

- Emulation of mobile device and infrastructure

- Device deployment

Integrating all those extra tools into the IDE saves setup and development time

Eclipse is designed to be extensible, all those features are implemented as plug-ins



TmL – Tools for mobile Linux

Introduction

What is TmL?

Why is TmL important?

TmL technology

Roadmap

Commercial uses of TmL

Demo



TmL – Tools for Mobile Linux

A sub-project of the Eclipse DSDP (Device Software Development Project)

Incubated in December 2006 with a 0.1 release slated for September 2008

Driven largely by Motorola, Inc., with contributions from other mobile and embedded developers

Goal is to provide support for development of mobile Linux applications on the Eclipse IDE using C/C++



Why TmL is Important to Linux Mobile Developers

Device development is hard enough (simplify)

Implementation of useful tools (gratis)

Source code available for inspection/alteration (libre)

Being used in commercial products today (stability)

TmL team understands the needs of the mobile and embedded developer (community)



TmL Technology (2008)

A framework for defining the types of devices that can be simulated/communicated

A mechanism for defining the states of execution and the transitions between those states

Reference implementation provides a VNC Viewer plugin to Eclipse from a Linux or React target OS

Available today on Eclipse CVS servers

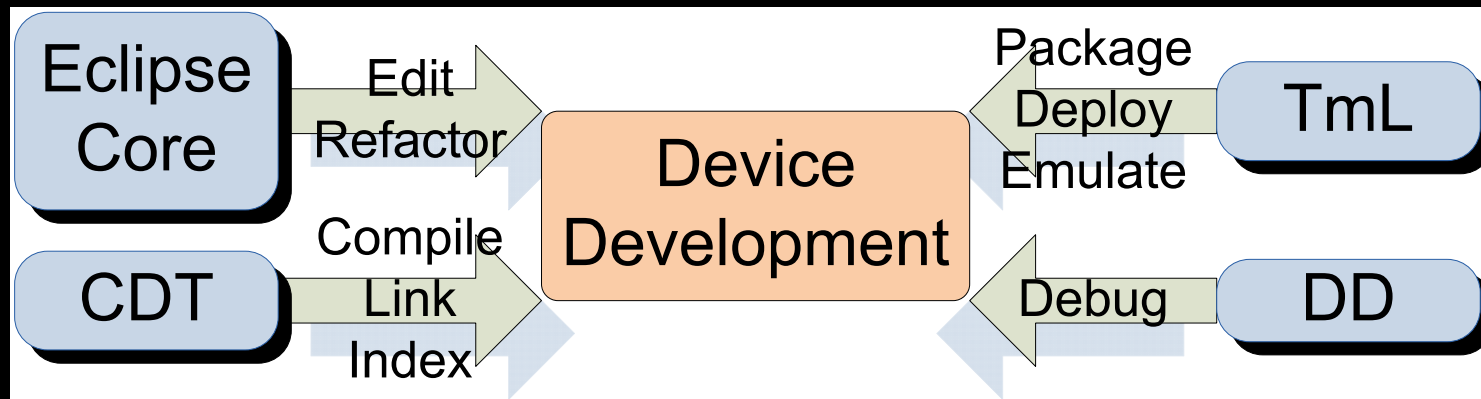


End-to-End Development for Linux Apps (2008)

Eclipse Core + CDT provides editing and compilation

DD (Device Debugging) provides breakpoint management, register reading, stack crawl, etc.

TmL provides deployment, connection, emulation, and interaction between the user and the running target



TmL Technology (2008 and beyond)

Simulate Specialized on-device Hardware

Camera

GPS

FM Radio

Orientation sensors

Simulate Infrastructure

Messaging (SMS/MMS)

Network (3G, Wi-Fi)

Broadcast services



TmL Technology (2008 and beyond) cont'd

Bring desktop development tools and methods to device development by integrating, contributing and extending other Eclipse projects

- Performance profiling

- Memory analysis

- Event logging

- Live and post-mortem analysis

Deployment to Linux-based...

- Development boards

- Emulators

- Mobile devices

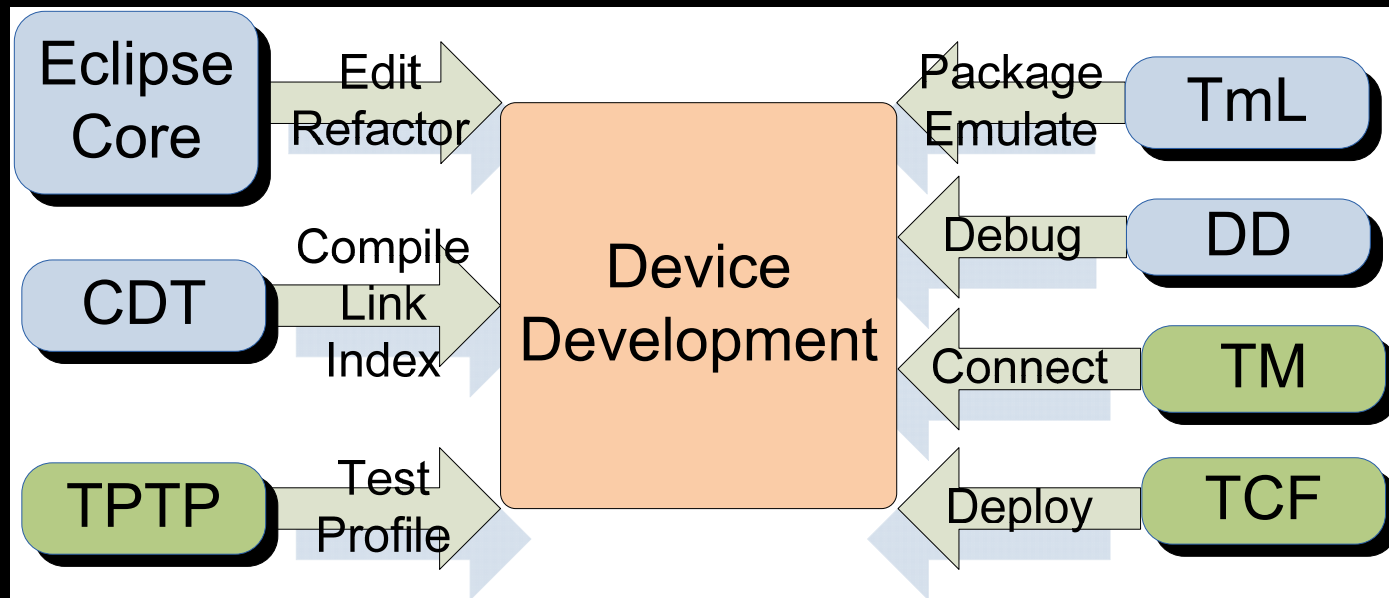


End-to-End Development for Linux Apps (Beyond 2008)

TM (Target Management) provides connection to the target device

TPTP (Test & Performance Tools) provides diagnostics, reports, and test

TCF provides data transfer and application deployment



TmL Roadmap

0.1 Release “soon” (September 2008)

Current stable functionality (State Machine Engine + VNC Viewer)

0.1.1 following Eclipse 3.4.1 (~October 15, 2008)

Bug fixes + integration

0.1.2 following Eclipse 3.4.2 (~February 1, 2009)

Bug fixes + integration

0.2 on the Eclipse Galileo train (June 25, 2009)

Device proxy framework + GPS reference implementation

Architectural refactoring

/proc filesystem viewer(s)



Demos

- Standard Eclipse + TmL VNC Viewer
- MOTODEV Studio for Linux



TmL Project Resources

Project web site:

<http://www.eclipse.org/dsdp/tml>

Project wiki:

<http://wiki.eclipse.org/DSDP/TML>

Regular phone meetings:

<http://wiki.eclipse.org/DSDP/TML/TmLOpenPhoneMeetings>

Developer mailing list:

<https://dev.eclipse.org/mailman/listinfo/dsdp-tml-dev>
dsdp-tml-dev@eclipse.org

TmL demo:

http://wiki.eclipse.org/DSDP/TML/How_to_configure_TmL_demo



TmL Committers and Contributors



Christian Kurzke
Architect



Eric Cloninger
TmL Project Lead



Daniel Franco



Otavio Ferranti



Fabio Fantato



Yu-Fen Kuo

Eugene Melekhov



Recruiting

Participate

Bi-weekly phone meetings (Tuesdays at 17:00 UTC)

1-877-825-8522 x1158163410

Contribute

Need developers for...

Device connectors/drivers for camera, orientation sensors and other hardware

Implementations of the /proc filesystem

Integration of TPTP features

Documentation & Sample Code



Questions & Answers

