

Integration of Development Tools for Automotive Software Development

Yuzhong Shen

Yuzhong.Shen@kuglermaag.com

Stuttgart, 10/06/2009

KUGLER MAAG CIE

Facts

- Today an high performing team (av. age of 44), acknowledged experts in their fields

Partners

- Partner of SEI/US, Sponsor of SEI-Europe
- Co-founder of iNTACS
- Driving Automotive Eclipse



Customers

- Global players, culturally diverse, operating in
 - Europe,
 - North America, and
 - APAC

Industries

- Automotive
- Transportation
- Finance / ICT
- Healthcare

Mission

Support customers in **mastering risks** associated with developing, acquiring, or delivering software, systems, and services while **maintaining the speed of innovation.**

Vision

First port of call for software, system, and service performance improvement – **in our selected markets.**

Eclipse Automotive Interest Group

Objectives

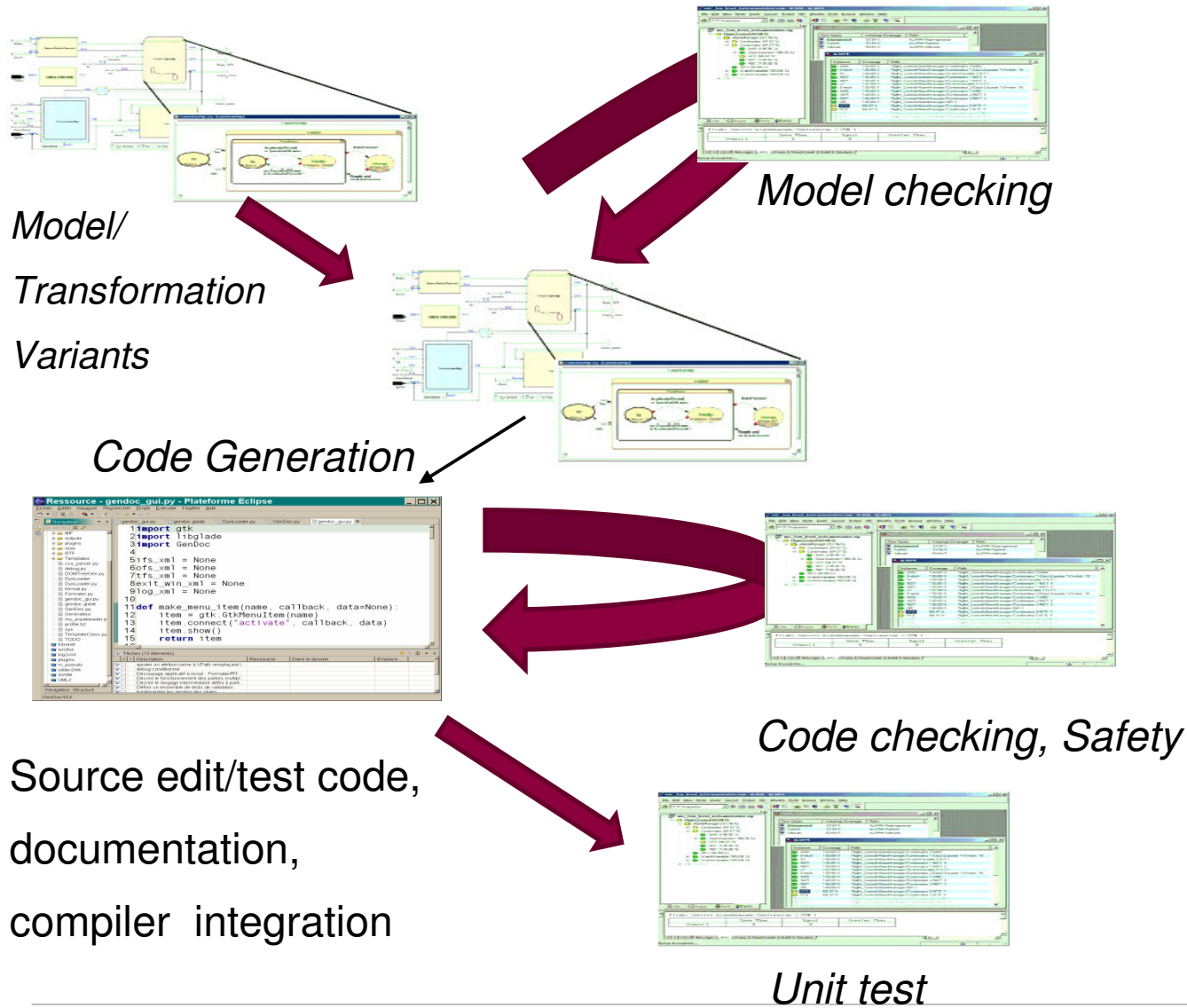
- To provide an infrastructure for tool development required by the automotive industry
- To address and support the needs for the whole automotive software development cycle
- To avoid that the same non-competitive basic tool functionality is redeveloped over and over again
- To join forces and meet current and future requirements in terms of tool runtime performance and memory consumption

Activities

- Definition of collaboration mode and structure of this IWG
- Provision of an Eclipse distribution for Automotive Tool Developers

http://wiki.eclipse.org/Auto_IWG

Automotive Software Development



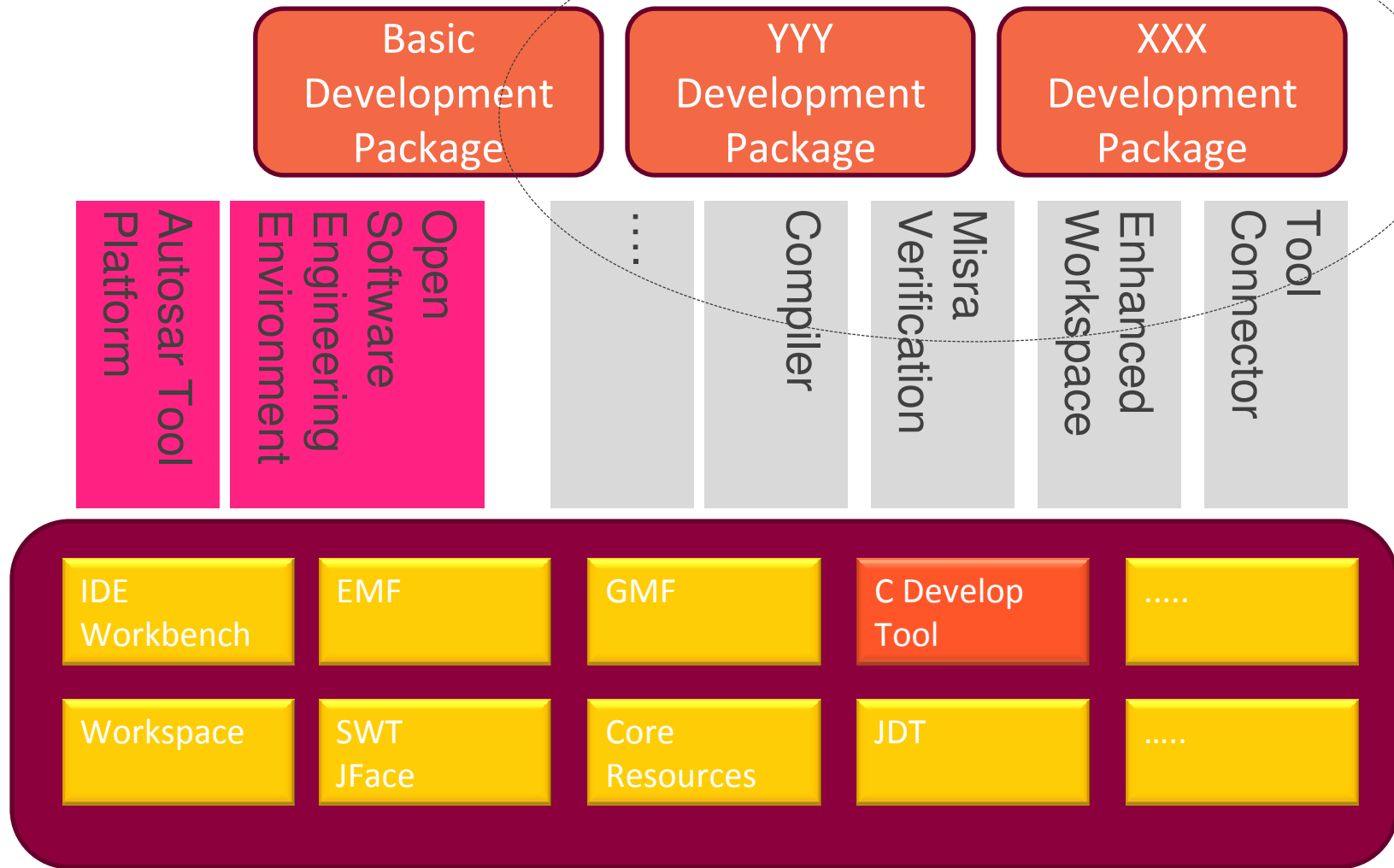
Eclipse Automotive Interest Group

Common Wishes

- Broad support of MISRA-C in the components
- Target Device emulation / Simulation
- Strategy for a model repository / distributed collaboration aspects
- Support for Component testing and Code Coverage Analysis
- Full Debugger Support
- Build management support
- C preprocessor enhancement
- Product line support / Variant management.
- C meta model that includes preprocessor statements.
- C++ (ANSI-C++ and/or embedded specific dialects like Extended Embedded C++)

http://wiki.eclipse.org/Auto_IWG

Possible Eclipse Automotive Package of Development Tools



Eclipse Automotive Interest Group and CDT

- CDT is a core enabling component for Eclipse Automotive Interest Group
 - Understand CDT
 - Develop tools on the top of CDT
 - Integration with CDT
 - Interworking with CDT
 - Common Requirements to CDT
 - Contribution to CDT
-
- http://wiki.eclipse.org/Auto_IWG





CDT State of the Union

Introduction for the Automotive Working Group

Doug Schaefer, Wind River
Eclipse CDT Project Lead

BTW: CDT == C/C++ Development Tooling

Where it all began



- Started by development team at QNX
- Contributed their C/C++ IDE to Eclipse in 2002
- Kick off meeting in July 2002
- CDT 1.0 released Nov 2002 based on Eclipse 2.0
- Initial contributors to CDT soon included Rational Software in early 2003
- Community grew from there



Who is the CDT

- 19 active committers
 - Most of who are no more than half time
- Representing major adopting vendors as well as user communities and a university (HSR)
 - Vendors: Wind River, IBM, QNX, Nokia, (Intel)
 - Users: Google, Ericsson, Broadcom, (Siemens)
- Numerous contributors of patches and to the newsgroups and mailing list
- Huge user community
 - 675,000 downloads of C/C++ IDE Package (SR1)



What is CDT

- Edit
 - Source Navigation between different points in code
 - Full featured editor for writing code
- Build
 - Integrate with external builder, like make
 - Managed build to let CDT do build
- Debug
 - Traditional Debug UI
 - Integration with external debugger

CDT Editor



- Extends JFace Text Editor
- Syntax coloring, extensible keyword list
- Bracket matching
- Inactive code (e.g. `#ifndef`) highlighting
- Optional Code Folding
- Text hover, extensible
- Content assist from parse/index and templates
- Refactoring

CDT Source Navigation



- CDT Indexer
 - Builds database of symbol defs and refs
 - Updated as files saved
 - “Fast” algorithm uses existing database to find defs in included headers
 - “Full” algorithm does full parse (not recommended)
 - Ability to include pre-built indexes for SDKs
- Open Declaration (F3 and Ctrl-Click)
- C/C++ Search Actions and Dialog



CDT Build

- Two modes, external and managed
- External builds calls out to external utility
 - User provides the build definitions file
 - Typically calls out to make, user provided Makefiles
- Managed build based on toolchain definitions
 - Generates UI elements for build settings
 - Generates Makefiles or uses it's own internal builder
- Most vendors provide their own mode

CDT Debug



- Typical Debugger Front End UI
 - Debug View for processes, threads, stack frames
 - Variables View
 - Expressions View
 - Breakpoints View
 - Memory View
 - Disassembly View
 - Registers View



CDT Debug Integration

- Currently two interfaces available
- C Debug Interface (CDI)
 - Original standard debug model
 - CDI/MI layer supports multiple gdb releases
- Debug Services Framework (DSF)
 - Services oriented interface
 - Asynchronous API to allow for slow connections
 - New and currently only supports gdb 6.6 onward

CDT 6.0 New and Noteworthy



- The Debug Services Framework has completed move to CDT and is a new component of CDT
- New heuristics to help indexer find header files in projects
- Added index support for implicit references and overloaded operators
- Improved Convert to C/C++ Project to factor in project types (e.g. Makefile)

CDT 6.0 New and Noteworthy



- New Launch Group launch config for launching multiple sessions at once
- New features for embedded development
 - Remote Launch based on RSE
 - GCC Cross compiler build support
- p2 support for installing tar files for C/C++ SDKs



What's Next

- Build System implementation needs an overhaul.
 - Especially Scanner Discovery
 - Scalability of managed build model
- DSF/GDB integration needs to reach parity with CDI/MI gdb integration
 - Then we can move to DSF for user community



What's Next

- Nokia investing in an integrated debugger
 - Replacement for gdb
 - Targeting Windows and Linux
 - Windows using Windows debug API
- On going maintenance work
 - Over 1250 bugs still open on CDT
 - Refactoring needs to be exercised more

Why is CDT Successful



- Supports build and debug tools from any vendor for any target on all major hosts
 - Including hardware debug tools
- Superb source navigation features
 - Open declaration, search, content assist
- It looks like the rest of Eclipse
 - And plays nicely with other plug-ins (e.g. Mylyn)
- It's free and everyone needs one



Why do we need your help

- Lots of code still to write
- Make sure CDT works for you
 - Don't count on others
- Help support user community
 - Especially Windows and Linux desktop
 - Help users on the newsgroup
- Contributions also needed for docs and test
 - Help improve quality for everyone

Thank you!



- Questions?