Gene-Auto development status and support

Andres Toom
IB Krates, Estonia
23/09/2009

www.krates.ee
Status after the Gene-Auto ITEA project
WP2 objectives

- Toolset architecture definition
  - Achieved
- Design and implement functionality for code generation
  - Achieved
- Design and implement functionality for formal model verification
  - Withdrawn from user requirements
- Ensure DO178B/ED12B-compliant development process and keep lifecycle data record for qualification
  - Partly achieved
Gene-Auto toolset

- One-step code generation from Simulink, Stateflow and Scicos models
- Open customisable architecture
- Open-source toolset to ensure long-term maintainability
- Generates ISO/IEC 9899 and MISRA compatible C code for embedded systems
- Usage of formal methods in selected transformation steps (WP5)
- Open intermediate languages for model exchange.
Simulink support

- Selected subset of supported blocks
  - 42 native Simulink blocks, 15 custom blocks
  - Easily extendable
- Support of multirate models
- Support of explicit scheduling via ‘function-call’ triggering
- Native support for matrix and vector operations
- Limited EML (Embedded Matlab) support in expressions
  - EML blocks not supported
  - Matlab functions not supported
- Modelling restrictions apply to ensure compatibility and chosen quality rules (D1.13)
Stateflow support

- Code generation from Stateflow
  - charts – supported
  - graphical functions – supported
  - “classical” truth tables – supported
  - EML (Embedded Matlab) truth tables – not supported
  - EML functions – not supported
- Modelling restrictions apply to ensure safety and chosen quality rules (D1.14)
**Scicos support**

- Scicos/Gene-Auto interface implemented in Scicos
  - Specific Scicos pallet compatible with the supported subset of Simulink blocks
  - Full user interface integration
  - Automated simulation support of the generated code
- Gene-Auto launcher
  - Reads the Scicos model stored in the GASystemModelling language and executes the required Gene-Auto elementary tools
“Qualification kit”

- Development plans
- Development data
  - High-level: Toolset requirements
  - Low-level: Tool requirements (each elementary tool)
  - Design
  - Source code
- Verification data
  - Requirement verification data
  - Design verification data
  - Code verification data
- User documentation
- Templates for qualification plan
Developments in 2009
Developments in 2009

- Maintenance for Airbus France and EADS Astrium
  - 20 support tickets / 25 technical tasks + related qualification data updates
  - No major features added

- Ada language backend with AdaCore
  - Specification and implementation of a new elementary tool and code generation chain
  - Sideresult: refinement of the tool requirements of TCPrinter
  - Sideresult: unused context argument elimination in the C-chain (potential)

- Other developments (IB Krates)
  - Extended testing framework being developed
  - Some technical tasks carried out on own account
Support for Gene-Auto users
Support for the community (services and actors)

- Public version of Gene-Auto
  - Since the beginning of 2009 with Gene-Auto v2.4.2
  - GPL licensed
  - Freely downloadable in source and binary forms
  - Maintained by IB Krates, Alyotech and FeRIA

- New features and bug-fixes
  - Funding by former consortium members (Airbus, Astrium, FeRIA, IB Krates)
  - Development from new projects (AdaCore, IB Krates)

- Integration of external contributions

- Public releases 2-3 times per year
Support for the community (channels)

- Support provided through the GForge website
  - Public forums
  - Public mailing list
  - Public tracker
  - Public documentation (published papers, user requirements, toolset requirements, user manuals, tool qualification plan user template, public case studies …)
  - Public releases
- www.geneauto.org
  - Frontend to the GForge collaborative site – highlights essential information and guides to detailed information either in GForge or Gene-Auto Pro site
THE GENE-AUTO PROJECT

Gene-Auto is an open-source toolset for real-time embedded systems. The toolset takes as input a functional description of an application specified in a high-level modelling language (Simulink/Stateflow/Scicos) and produces C (in close future also Ada) code as output.
Commercial support

- Dedicated development and support contracts
  - IB Krates, Alyotech
- Gene-Auto Pro
  - geneauto.krates.ee, IB Krates
  - Flat-fee based maintenance scheme
  - Access to the latest developments (development snapshots and releases)
  - Knowledgebase, FAQ, detailed user manuals
  - E-mail and phone support
  - Task database (Customised front-end for gPM) (not yet available)
  - Vote on CCB (Change Control Board)
  - Testing framework and extra tools (not yet available)
How do I change the way code is generated from a supported block?

Submitted by Tõnu Nõks on Fri, 14/08/2009 - 12:23

Changing the block implementation can be done exactly the same way as defining new blocks (see How do I add support for new blocks?). You implement the backend and typer pair or a pre-compiled library function corresponding to the block’s semantics and provide a custom block-library configuration file when launching the tool.

When loading block type descriptions the toolset always first processes the standard library and after that the custom library. If a block is defined both in the standard library and in the custom library, then the definition from the custom library takes precedence.

Is there any support for composing and processing Gene-Auto models and data types in custom tools?
Gene-Auto PRO subscription levels

- Gene-auto Pro registered user (free)
  - Access to knowledgebase (public)
  - Access to error reporting module (read only)

- Subscriber silver
  - Access to knowledgebase (pro+public)
  - Access to latest releases and development snapshots
  - Access to documentation
  - Access to error reporting module (read-write)

- Subscriber gold
  - Vote on CCB
  - Unlimited e-mail support

- Subscriber platinum
  - Dedicated budget
  - Phone support
Change Control Board (CCB)

- All changes are coordinated by the CCB
- CCB is a virtual body composed of
  - End users with active maintenance contract
  - Gene-Auto PRO subscribers
  - Representatives of developers
- CCB meetings
  - Determine the priorities of toolset development
  - Approve, postpone or reject proposed changes
Future

- Ada backend
  - Estimated beta version by the end of 2009
- Potential additions
  - SysML importer, Simulink exporter,
  - Support for a subset of the Matlab language
  - Verification tools
  - Optimisation, target adaptation
- Supporting infrastructure
- Qualification data preparation
- Increasing the robustness of the tool
- Growing the user base
Thank you!

www.geneauto.org

geneauto.krates.ee