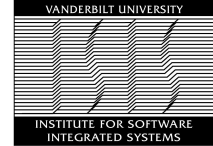


ECSL-DP: Embedded Control Systems Modeling Language and Toolchain for Developing High-Performance Embedded Automotive Applications¹

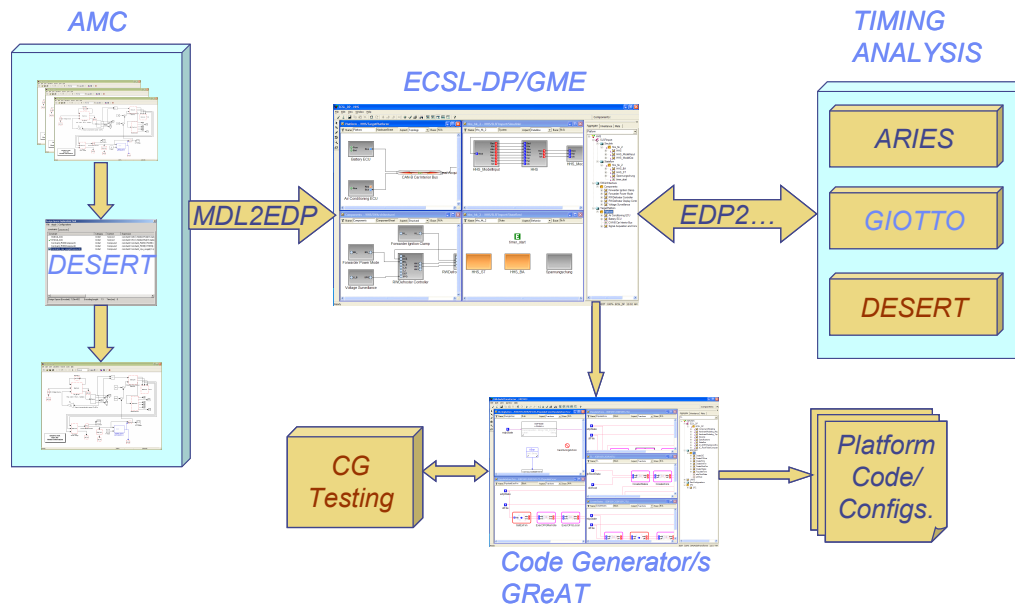


Summary of Features

ECSL-DP is a domain-specific modeling language to model and construct embedded automotive applications running on a distributed platform including multiple Electronic Control Units (ECU-s), connected with a Time-Triggered (TTP/FlexRay), Controlled Area Network (CAN/LIN) bus. The language is supported by a set of integrated tools that fully support a model-based design process: component modeling, system architecting and modeling, automated model compilation using design-space exploration, behavior and configuration code synthesis and execution.

The ECSL-DP Toolchain implements a model-based design process. The toolchain includes the following elements (as illustrated in the figure below):

- Functional and Behavioral Modeling Tool (Mathworks® Simulink/Stateflow) support creation of functional model of control applications using the continuous and discrete behavioral modeling capabilities of the Mathworks toolsuite.
- Automated Model Compiler using DESERT that automatically composes a model from a set of sub-models and an architectural description of the arrangement of sub-models, ensuring full-connectivity of all control flow and data flow signals between sub-models, proper sequencing of sub-models, and compatibility of sub-models
- Component and System Modeling Tool (ECSL-DP/GME visual model editor) that supports componentization of functional models, system architecture models which describe component interactions, hardware platform models that describe the embedded platform – ECU-s, buses, component allocations, and task mapping etc.
- Design Analysis Tools (AIRES/DESERT/GIOTTO) that supports event and timing analysis (including schedulability), and static scheduling of tasks, and communication over time-triggered bus.
- Code Generation tools for behavioral (Stateflow to C-code) and configuration code synthesis, developed using a declarative graph rewriting technique implemented in the GReAT tool
- Tools for testing of code generators that have been specified with GReAT.



Vehicle Control Platform Tool-chain

The tools are integrated using an Open Tool Integration Framework that allows including other tools into the toolchain if needed.

Contact point: Sandeep Neema, sandeep.neema@vanderbilt.edu, (615) 343-7472, ISIS, PO Box 1829B, Vanderbilt University, Nashville, TN 37235.

¹Original research and development was supported by DARPA/IXO MOBIES program through USAFRL.