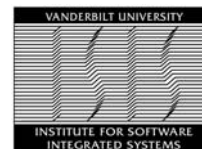


GRaT: Graph Rewriting and Transformations Model Transformation Environment for Model-Based of Systems¹ Summary of Features



Model-based development of systems, especially embedded systems, requires technology for transforming models. Implementing model transformation tools by hand is an error prone process, and even newer technologies, like XML transformation scripts (XSLT) are difficult to use and inefficient on large and complex models.

GRaT is a graph-transformation based language that supports the high-level specification of complex model transformation programs. In this language, one describes the transformations as sequenced graph rewriting rules that operate on the input models and construct an output model. The rules specify complex rewriting operations in a concise, yet precise manner, in the form of a matching pattern and a subgraph to be created as the result of the application of the rule. The rules (1) always operate in a context: a specific subgraph of the input, and (2) are explicitly sequenced for efficient execution. The rules are specified visually using a graphical model builder tool.

GRaT is supported by a toolchain that includes:

- The GRaT *Modeling Tool* to construct model transformation programs
- The GRaT *Engine* that directly interprets and executes GRaT programs
- The GRaT *Debugger* that provides an interactive environment for debugging GRaT programs
- The GRaT *Code Generator* that compiles GRaT programs into efficient C++ code
- Supporting *run-time modules* for generating output text from graph datastructures (produced by GRaT)

The tool chain is fully integrated with the Generic Modeling Environment.

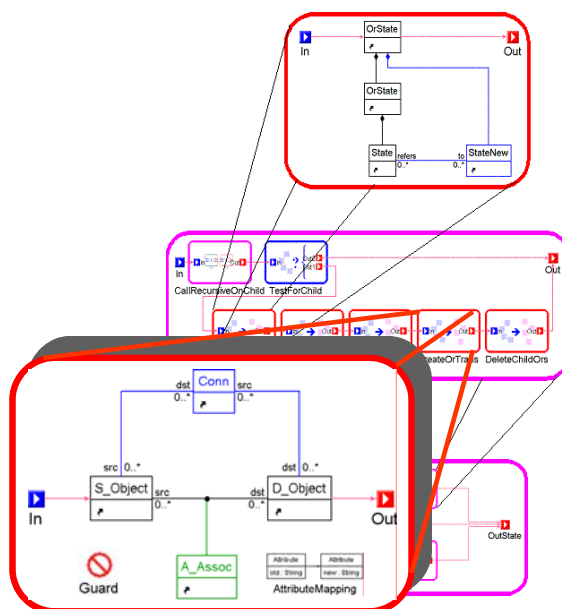
GRaT has been used in developing various complex model transformation programs, including:

- Code generator for Simulink/Stateflow models
- Transformation tool for converting Simulink/Stateflow models into hybrid automata models
- Model translators for toolchains supporting embedded system development.

GRaT is also applicable as a tool for supporting MDA development, as it provides most (but not all) the capabilities prescribed by the OMG QVT specifications. As such, it can be embedded into domain-specific, model-based software development toolchains to provide direct support for domain-specific modeling languages and architectures. These toolchains could then be used to construct embedded software for C2 systems, avionics, robotic systems, and the like.

Contact point: Gabor Karsai, gabor.karsai@vanderbilt.edu, (615) 343-7472, ISIS, PO Box 1829B, Vanderbilt University, Nashville, TN 37235.

Example GRaT Program



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