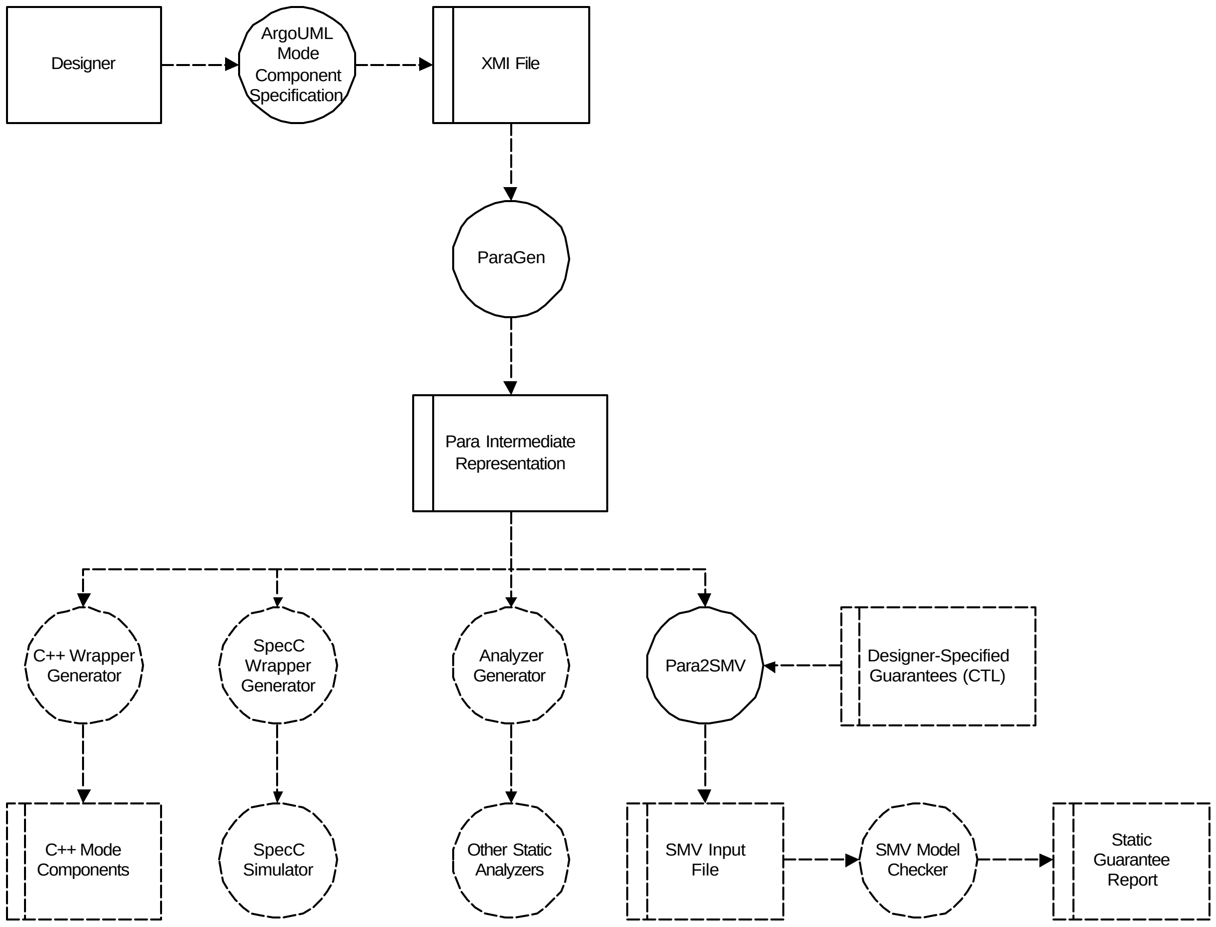
DYNamic Assembly from Models

The DYNAMO project is concerned with assembling high-assurance systems from components, and, specifically, with guaranteeing correct interaction of sets of large, heterogeneous components. Several problems must be overcome to provide such guarantees: 1) dealing with the sheer complexity of the individual components and their interoperation; 2) maintaining design integrity and information hiding in the individual components; 3) providing the desired guarantees, either statically, during design, or at run time; and 4) not compromising efficiency while accomplishing the other goals.

DYNAMO addresses these problems with several techniques: 1) a layered architecture limits complexity by reducing the quantity and nature of allowed interactions; 2) a declarative specification mechanism abstracts away low-level details such as event dispatch and handling and variable updates; 3) static analysis of component designs is augmented with run-time status monitoring and gauges; and 4) compile-time component wrapper generation removes expensive, inter-layer procedure calls.

DYNAMO Tools



Georgia Institute of Technology

Michigan State University

- **ArgoUML**: ArgoUML is a UML design tool originally developed at the University of California at Irvine and now available from URL http://argouml.tigris.org. Using it, designers specify mode components and their composition into an assembly. In particular, the classes in a UML class diagram denote mode components, and the associations between them denote intercomponent layering. OCL annotations of the associations denote intercomponent constraints. Moreover, each class may have a corresponding state chart that specifies its synchronization behavior.
- **Xerces**: ArgoUML files are saved in XMI format. XMI is an XML DTD used to describe data from CASE tools. Xerces is an XML parser that can be obtained at URL http://www.apache.org. DYNAMO uses Xerces to implement the front end to ParaGen.
- **Para**: XMI is verbose and difficult to access. Para is an intermediate representation and API that provides easy access to ArgoUML designs. It can be used independently of DYNAMO as a representation for any tool requiring access to designs expressed in XMI.
- **ParaGen**: ParaGen is a translator for converting XMI into the Para intermediate representation. It uses the Xerces XMI parser to analyze XMI designs. An API is provided to enable tools to access designs in Para format. ParaGen can also produce output into a text file for external perusal or scripted post-processing.
- **Para2SMV**: DYNAMO provides design and run-time guarantees of synchronization behavior of assemblies of components. One example of design analysis is model checking. Para2SMV is a tool for converting designs expressed in Para into the input format of SMV.
- **SMV**: SMV is a symbolic model checker originally developed at Carnegie-Mellon University and available at URL http://www-cad.eecs.berkeley.edu/~kenmcmil/smv. It requires two sources of input: a state machine model and a guarantee to be checked. In DYNAMO, design models are converted into SMV state machines using Para2SMV. Guarantees are expressed in CTL (Computation Tree Logic) and are provided by the designer as a separate input to Para2SMV.
- **SmvModel**: SmvModel is a collection of C++ classes that can be used to help generate SMV input files. They are used internally to DYNAMO by Para2SMV, but may be used by any tool wishing to generate SMV input files.

Mode Components

A *mode component* is a hierarchical software component whose interface provides a continuously updated view of its current status. Clients of a mode component can interact with it at a high level of abstraction, thereby supporting correct synchronous composition. A mode component provides status information to components above it in a layered architecture. The state of those components is updated automatically when changes occur to any of the status information in the mode component. The nature of the update is expressed declaratively without requiring detailed specification of event synchronization. Client components can also make service requests to a mode component using traditional method calls.

Mode components are implemented using C++ templates and mixin layers to wrap traditional class definitions. The templates are generated automatically from a design specification provided in ArgoUML. Use of templates and other advanced C++ features implements the specification without compromising the performance of the assembly of components.

Contact Information

dynamo-support@cc.gatech.edu
http://www.cc.gatech.edu/dynamo

Sponsorship

Effort sponsored by the Defense Advanced Research Projects Agency, and the United States Air Force Research Laboratory, under agreement number F30602-00-2-0618.

Georgia Institute of Technology

Michigan State University