Final Report

Active Models in Support of Collaborative Design

Contract Number: N00014-96-1-0699

Reporting Period: 04/01/1996-12/31/1999

Submission Date: April 17, 2001

Prepared by:
Robert L. Constable
Cornell University
4130 Upson Hall
Ithaca, NY 14853
"Active Models in Support of Collaborative Design"

Robert L. Constable

Cornell University
4130 Upson Hall
Ithaca, NY 14853

Ms. Susan Aylward
Office of Sponsored Programs
123 Day Hall
Ithaca, NY 14853

Approved for public release

The project was concerned with providing mathematical models of artifacts that capture both their externally visible properties and their behavior as well. It was also concerned with providing ways to share these models among various software systems.
Final Report

This project was concerned with providing mathematical models of artifacts that capture both their externally visible properties and the behavior of the artifact. It was also concerned with providing ways to share these models among various software systems. The main innovation from this project was the extensive development of a language for combining active models. The language was called the MathBus based on a formulation in Collaborative Mathematics Environments [4]. This language was designed to enable sharing the active models with software systems needed to explore them, such as symbolic algebra systems, problem solving environments (such as Matlab), and theorem provers such as PVS and Nuprl.

The project also supported work on automatic mesh generation. This was a particularly important component of a problem solving environment of the kind that Richard Zippel and Paul Chew were developing for the project. This mesh generation work has continued in vital ways at Cornell, supporting many other projects.

The MathBus [1] design has proven effective in communicating mathematical data across differing systems and platforms. Two such instances of its applications include the Nuprl logical programming environment [15] and Jprover [16], an intuitionistic theorem prover that has been integrated with the Nuprl and MetaPRL [2] proof assistants. In both of these examples, the MathBus term structure is used as a means for communicating terms across different systems with inherently different term syntax.

The MathBus design is based on a tree structure that is independent of any programming language, making it well suited for inter-process sharing and utilization of mathematical theorems, definitions, and proofs, which are difficult to represent in common ASCII text. The basic MathBus term is created by specifying an identifier for a given node, and also subterms for that node, which may be either MathBus terms or simple types, such as integers or strings. This tree structure created is then converted, with the aid of a shared registry, into a series of 32-bit integers that comprise the binary byte stream, which at last gets encoded using the MIME (RFC 1563).

MathBus has been implemented in languages including LISP, Ocaml, and C++, and implementations in other languages can follow easily from its design. We include in MathBus not only the commonality of syntax, but also facilities for optimizing transmission, hash-coding mechanisms, variable binding mechanisms, and more.
References

   www.cs.cornell.edu/simlab/papers/mathbus/mathTerm.htm


