AN ANALYST-ORIENTED DEVELOPMENT APPROACH
FOR REAL-TIME ADA SOFTWARE
T.R.Nicholas, D.P.Cichy, J.Spears
Martin Marietta Astronautics Group
P.O. Box 179 Denver, CO 80120
(303) 971-9957

Abstract
We introduce an approach for developing real time Ada software systems, called Interactive Ada Development Approach (IADA). The approach features automated translation of a graphic model to code and a unified, analyst-centered environment for software development. The IADA demonstrates an enhancement of the state of the process and supporting technologies which will allow Ada to be produced for deliverable real-time applications from a graphic analysis using an enhanced commercial CASE tool and other standard utilities. The discussion also introduces the notion of the model-centered analysis which greatly simplifies the task of managing and maintaining a software product.

Introduction
The efficient, cost effective development and fielding of Ada software for real-time applications such as satellite and robot control systems has been considered to be difficult at best. With the development of object-based methodologies and technologies, the gap between a workable prototype and a operational system in Ada has been closed somewhat. Recent developments in Object-Oriented Analysis (1,2), Recursive Design (3) and Structured Rapid Prototyping (4) have shown good promise in enhancing the productivity of requirements analysis tasks. To date, however, a comprehensive (i.e. covering the entire software life cycle) approach using these methods has not been proven.

The approach summarized here includes significant extensions and enhancements to the object oriented methods which forms its basis. The result is an analysis-centered approach, focused on real time Ada applications such as robotics and space command and control systems, which covers the entire software life cycle.

The IADA
The Interactive Ada Development Approach (IADA) consists of a customized methodology, a model-centered technology and a unique application of commercial and proprietary software tools. In place of the traditional (i.e. waterfall) life cycle the IADA methodology defines three phases of system definition called Views: the Application View, the Host View and the Product View. The products of the IADA for each view include a graphical/textual model set, executable Ada software and an automatically produced document.

The Application View represents a specification of the software system in the application domain. The Host View introduces features of Software Engineering domain such as class structures, inheritance and reuse. The product of the Host View is functionally complete software which can execute in an unconstrained (host) environment. The Product View introduces optimization structures and hardware architecture features in the architecture domain. The software produced by the Product View is executable on a target computer, and is suitable for delivery.

The three Views may be produced in parallel since each one emphasizes a different discipline. Each view is constructed recursively following the principles of rapid prototyping; the system is modeled in increasing detail over time while involving the customer (user) in the process through informal and formal walkthroughs.

The IADA features a translator which produces executable Ada software directly from a single set of graphic models. The software produced by the translator is optimizable from the model set.

The Model-centered Environment allows analysts from different disciplines to work with a single integrated set of models, each producing software and documents which are directly derived from the analysis. The IADA requires changes to be made to the model, not to the code, thereby simplifying the configuration management and traceability aspects of the developed system. Integration is accomplished using a "model merge" feature of the CASE tool. Using IADA, analysts can derive requirements and design architectures and perform optimization analyses using a single, integrated set of software tools and a unified approach. The same approach and tools produce controlled, deliverable software and documentation.

References