

# The (continuing) Evolution of UML for Real-Time Systems Development

"Please be advised that, as of now, Mr Alan Moore's column will appear in every issue of the magazine"

## UML - A BRIEF HISTORY

During the early 1990's, many individuals and organizations experimented with object-oriented concepts, notations, methods, and languages. This created a wide and varied proliferation of techniques used to build applications with this new technology. The "method wars" looked like they might stall the adoption of these improved engineering practices through simple confusion. In an effort to consolidate the industry, three key methodologists (Grady Booch, James Rumbaugh & Ivar Jacobson - "the three amigos") formed a coalition to produce a Unified Method. Over the course of the past four years, others have joined this effort under the auspices of the Object Management Group (OMG), and we now have an industry standard Unified Modeling Language (UML) soon to be at version 1.4 maintained by the OMG.

The UML provides an excellent starting point for defining and designing computer systems, but for real-time development, the UML needs to go further. Recognizing this, the OMG has spawned a new initiative, the Real-time Analysis and Design Group (RTAD), specifically to recommend UML extensions in this area.

The goal of the RTAD initiative is "to issue a comprehensive set of RfPs (Requests for Proposal) leading to

OMG standards that will support the use of object-oriented approaches in the analysis, design, and development of real-time computing systems". There have been three RfPs proposed by the RTAD:

- UML Profile for Scheduling, Performance, and Time (OMG document ref: ad/99-03-13);
- UML Profile for Quality of Service (QoS) other than timeliness (draft RfP, ref: ad/99-08-06);
- UML Profile for large-scale, distributed systems (no RfP as yet).

Only the first has received much attention, with an initial technology submission made and the final submission due in June 2001.

## SPECIAL ISSUES IN REAL-TIME SOFTWARE DEVELOPMENT

The development of real-time, embedded systems is a tricky business. The very nature of these applications makes certain characteristics of their implementation such as timing and implementation architecture critical. The following sections identify some key areas that need to be addressed as part of the systems analysis and design process for real-time system development.

## QUALITY OF SERVICE

System requirements often include Quality of Service (QoS) in particular specific timing constraints relevant to product performance. These constraints are captured and decomposed during the system definition

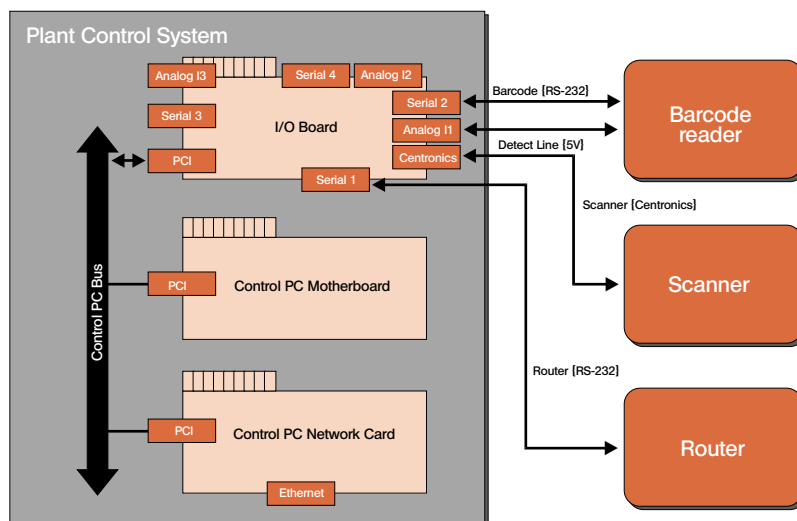


Figure 1. Plant control system.

stage in order to facilitate a reasonable system design and to support system test. The area of timing is covered in the initial submission of the RTAD Scheduling RfP, which was presented at the September OMG meeting.

## SYSTEM ENGINEERING

System engineers design at a high level and influence both hardware and software composition of the system. They are concerned with the overall architecture and usually make tradeoffs between implementing functionality in hardware, software, or both. They provide a vital coordinating role across the engineering team, and any modeling language that wishes to support real-time and embedded system development must provide adequate support for their needs.

The two most important deficiencies in UML from a system engineer's perspective are the poor support for data flow, and the difficulty in describing system architecture. System engineers often use data flow concepts, both to describe overall behavior, and to define the key algorithms in system control. The only tool that they currently have in UML is the Activity Diagram, which implements very few of the true dataflow concepts and is aimed at class design, rather than system design. They are often also concerned with accurately representing the frequently complex architecture of the total system, something that the current Deployment Diagram is woefully inadequate for.

Both of these areas are addressed in the UML 2.0 Superstructure RfP (ref: ad/00-09-02), which has recently been issued and is due for adoption in May 2002. As an aside, these two areas are also of interest to hardware engineers, although I suspect that use of UML in that domain is some way off.

## CONCURRENCY MODELING

There is an inherent concurrency in most real-time, embedded systems as they manage multiple I/O devices and control from a variety of interfaces. This is an area that not well specified in the UML Examples are:

- The exact semantics of active and passive objects;
- The relationship between passive objects and state-machines;
- The selective nature of the target object type for call and send actions (potentially violating the principle of encapsulation).

These problems have been aired recently in the discussions on action semantics (ref: ad/98-11-01), whose initial submission was issued in September (ref: ad/00-08-02). This RfP requires technology to describe the detailed semantics of actions, such as assignment, function calls etc. Unfortunately, the submitted technology is constrained to fit with UML 1.4 and so the submitters cannot make the changes that some (including myself) feel need to be made, and therefore will have to be left to UML 2.0.

## CONCLUSION

The OO modeling community has made great strides

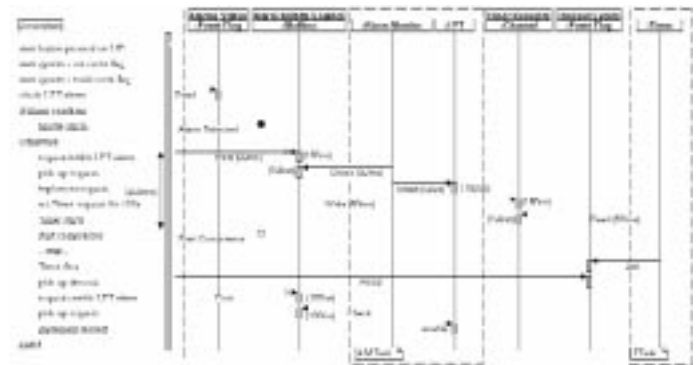


Figure 2.

in the last five years, spearheaded by the OMG with UML. However, only in the last two years has serious consideration been given within the OMG to the needs of embedded and real-time system developers. Although the current UML is sufficient for many real-time projects which are already using UML to great effect, there are three main areas that still require work: Quality of Service, system engineering support, and concurrency semantics. These are now all receiving significant attention and, with the release of UML 2.0, real-time engineers will finally have a modeling language that meets all of their most pressing needs ■

---

*Alan has 15 years of experience in the development of real-time and object-oriented methodologies, and their applications in a variety of problem domains. He has been actively involved in product development, training and consulting related to OOAD and structured development tools during that time. Alan has co-authored a book on GUI design and published several papers, and has lectured on a wide variety of analysis and design issues.*

*Alan is responsible for the specification, planning and management of the ARTISAN product strategy. He is the author of ARTISAN Real-time Perspective, a pragmatic approach to the development of real-time systems and is an active participant in the Real-time Analysis and Design Group (RTAD) of the Object Management Group (OMG).*