



Evolution of Real-time Standards in NCITS TC R1, Real-time Computing Systems

Russ Richards
Chair, NCITS TC R1, Real-time Computing Systems

Status Report to the Real-time and Embedded Systems Forum
The Open Group
Four Seasons Regent Hotel
Austin, Texas USA
18 July 2001



*National Committee for
Information Technology Standards*

**Chair, NCITS TC R1,
Real-time Computing Systems**



Assumptions about DOD Real-time Requirements

- DOD is required by PL 104-113 and OMB Circular A-119 to participate in the development of consensus standards to ***ensure that public and private sector needs are satisfied.***
- Consequently, when the words “needs” or “requirements” are used in this briefing the assumptions are:
 - The examples, though DOD-based, can and are extensible to the private sector.
 - Cost savings anticipated by such standards efforts will inure to the benefit of both the public and private sector.



Real-time Requirements: Development Assumptions

- **NEEDS:** There are a broad number of needs for real-time standards in both language-specific and non-language-specific environments
- **Vendors (and to some extent intellectual property rights) must be considered in many of the projects identified**
 - **Defining the nature of the work to avoid IPR issues can and should be accomplished prior to formal project proposals and subsequent standards work.**
 - **Vendors are generally willing to work with consensus standards bodies to resolve issues and define meaningful work for the standards bodies.**
- **Demonstration project and requirements documents are often available as a foundation for standards work.**
- **Community of interest buy-in to standards projects is essential**





History of NCITS and Real-time Standards

- National Institute for Standards and Technology (NIST) hosted a series of open workshops to chronicle Java real-time requirements [Summer 1998]
- A separate group, Real-time Java Working Group, formed to begin developing a proposed specification to satisfy the NIST Real-time Java Workshop requirements.
- The group approached NIST (Nov 1998) to propose a standards development project and the formation of a Technical Committee to support it (ala R1):
 - Sun Microsystems claimed the project would infringe upon Intellectual Property Rights
 - NIST recognized the general need for a Real-time computing standards committee and formed it in Dec 1998
 - Balloted the proposed real-time Java Project [failed]
 - DoD proposed 13 real-time projects; 4 were selected
- Real-time Java went two directions (J Consortium & JCP)



Outline of Real-time Requirements Presented to SDOs

Non-language Specific

- Architecture description language
- Device driver portability
- Portability testing
- Standard test suite structure
- Fault management
- Security for Real Time
- Plugable Protocols
- POSIX and Windows NT
- CORBA

Language Specific

- C++
- Ada
- Java



Projects Initially Selected by NCITS R1

RT Technical Reference Model (GOA) [Report]

Non-language Specific

- Architecture description language
- Device driver portability [Standard]
- Portability testing [Report]
- Standard test suite structure [Standard]
- Fault management
- Security for Real Time
- Plugable Protocols
- POSIX and Windows NT
- CORBA

Language Specific

- C++
- Ada
- Java



National Committee for
Information Technology Standards



Scope and “Volunteership”

- Many of the initial members of NCITS TC R-1 joined and paid dues based on the potential that R1 would propose the development of real-time extensions to Java as a project.
- When that proposal did not materialize, many dropped out.
- R1, however, has a much broader technology purview than extensions to a single programming language:
 - Some substantive real-time standards projects have been approved.
 - Many more are available to pursue as “volunteership” increases.
- Consequently, it is hoped that we can win back those who have been absent and add many more companies and organizations to the list of current member companies providing volunteers:
 - DoD/DISA
 - Lockheed Martin
 - Sun Microsystems
 - Hewlitt Packard
 - MPI Softtech
 - Farance Inc.





Projects Approved and Underway

- **R1, Real-time Reference Model**
 - **NCITS Project: 1372 - DT (Technical Report)**
 - **Title:** Real-Time Generic Open Architecture (RT-GOA)
- **R1.1, Real-Time Device Drivers**
 - **NCITS Project: 1373 - D (Standard)**
 - **Title:** Real-Time Device Driver Portability
- **R1.2, Real-Time Testing**
 - **NCITS Project: 1374 - DT (Technical Report)**
 - **Title:** Real-Time Application Portability Testing
 - **NCITS Project: 1376 - D (Standard)**
 - **Title:** Real-Time Test Suite Structures



R1 and Uniform Device Drivers (UDIs)

- **Uniform Driver Interface (UDI)** is a standard that is intended to make I/O device drivers and protocol modules portable between computer platforms and operating systems.
- The National Committee for Information Technology Standards (NCITS) has a technical committee, R1, whose purpose is to provide advice and additional documentation, including standards, which will improve the value UDI for real-time and embedded uses.
- This group has identified a number of issues relating to real-time and embedded UDI drivers and protocol modules.



Scope

- **The scope of real-time and its relationship to UDI includes applications**
 - **With custom security considerations and**
 - **Where performance requirements must be predictable (deterministic) particularly those that:**
 - **“Require” rapid response to an input**
 - **Are safety critical**
 - **Require a high degree of fault tolerance**
 - **Ensure quality of service**



Issue Categories: Processing Priority and Scheduling

- **Priority vs. Requesting Thread Priority**
 - Priority Inversion
 - Hidden Priority Inversion
 - Preemptability
 - Scheduling Classes
- **Status of Priority Issues**
 - The current approach is to be able to associate the processing priority of an I/O request with the priority of the requesting thread.
 - Project UDI has updated their standard to provide the underlying infrastructure to allow for this. See the UDI Specification Corrections Document, Amendment Core A20000407.3. This update adds an origin handle to control blocks.
 - An implementation of the UDI environment that supports real-time is expected to use a structure or variable referenced by this origin handle to indicate the priority of the requesting thread.



Issue Categories:

Issues not yet resolved

- **Issues not yet resolved**
 - **I/O Priority vs. Requesting Thread Priority**
 - **Priority Inversion**
 - **Hidden Priority Inversion**
 - **Preemptability**
 - **Scheduling Classes**
 - **Control Block Scheduling**
 - **Procurement Issues**



Issue Categories: Network Protocol Metalanguages

- **Status**
 - The metalanguage specifications are in a very early stage, but is progressing rapidly. The initial draft of the specification is being discussed with Project UDI.
 - This start has developed considerable interest. There has been considerable discussions at UDI meetings.
 - Project UDI has a weekly teleconference and an email reflector to continue these discussions at a faster pace. The phone calls are at noontime Eastern time.
 - You can sign up for the email reflector by going to www.sourceforge.net.
- **Issues not yet resolved**
 - Need to continue to define the metalanguages.



Issue Categories: Resource Allocation

- **Status**
 - These issues have been discussed by Project UDI. Project UDI is depending on R1.1 to identify real-time needs.
 - Deterministic behavior is not their concern.
 - They will support R1.1 needs.
 - R1.1 needs to define them.
- **Issues not yet resolved**
 - These issues have not been discussed by R1.1.
 - Either proposed solution may require a document, perhaps a specification.
 - There is a question of whether this should be a UDI specification, or something produced by R1.1.



Issue Categories: Mappings of POSIX APIs to UDI

- **Status**

- The POSIX APIs have been identified and reviewed. Two issues were identified.

- **The first is synchronized I/ O.**

- One of the purposes of synchronized I/O is to allow applications in specific portions of their code to know when the disk metal has been updated.
- Most if not all disks today as far as we know don't support this capability.
- If a disk were produced with this capability, it would make sense to use it only when a synchronized I/O request is received.
- To accomplish this the I/O driver would need to be able to distinguish between a normal write and a synchronized write.
- This type of functionality is not in the current SCSI metalanguage.

- **The second is the GIO metalanguage.**

- This is the most likely metalanguage that special I/O drivers will use.
- Since there is no standard specifying the mappings between applications and operating systems, this may be implemented differently on different operating systems. Other POSIX interfaces have been reviewed.





Issue Categories:

Mappings of POSIX APIs to UDI

- **Issues not yet resolved**
 - The synchronized I/O issue was discussed with Project UDI.
 - It was felt that most, if not all, applications use synchronized I/O to flush data to the disk at appropriate times.
 - UDI provides adequate support for this.
 - No application has been identified that needs to know more precisely when data actually gets into a state where it will survive failures. For this reason, it was felt that there is no need to make changes to the SCSI metalanguage at this time.
 - The GIO metalanguage has received considerable discussion within Project UDI.
 - Project UDI is currently defining how GIO works with read () and write ().
 - They are also providing a new application interface for GIO.
 - R1 and R1.1 should decide whether UDI behavior needs to be specified for other POSIX functionality.
 - The behavior such as asynchronous I/O can be implemented.
 - They are not required to be implemented, however, and they may not be implemented consistently.



Issue Categories:

SNMP Metalanguage

- UDI defines a set of statistics that the UDI environment can make available to an SNMP interface. There is no requirement for the environment to do so. The UDI metalanguages defines these statistics. It is desirable to have a generic SNMP metalanguage. The reasons for this are listed below.
 - SNMP provides an easy and standard interface to send commands to an I/O driver or protocol module even over a network. An example of such a command could be a configuration or reconfiguration request. A SNMP metalanguage will provide this capability.
 - The standard UDI statistics allows a program to obtain only the statistics supported by the metalanguage. Drivers may want to provide additional statistics. An SNMP metalanguage will permit this. This is especially useful for special drivers that use the GIO metalanguage. It is anticipated the most special drivers will use the GIO metalanguage.
 - The SNMP programming interface can be used by UDI development tools.
 - The SNMP programming interface can provide I/O driver debug capabilities.



Issue Categories: SNMP Metalanguage

- **Status**
 - The idea of an SNMP metalanguage was discussed with Project UDI.
 - In fact the idea originally came from a Project UDI member.
 - The metalanguage should be a simple one to specify.
 - No additional progress has been made.
- **Issues not yet resolved**
 - R1 and R1.1 needs to decide whether or not this is important.
 - If it is, it would probably speed things up if they proposed a metalanguage.



Issue Categories:

Driver Real-time Development Specification

- End users will want to procure real-time I/O drivers. They need to have a reasonable expectation of what they will get when they reference the specification. Currently there is no specification or guidance as to what I/O driver developers or UDI environment implementers need to do to qualify as a real-time implementation.
- **Status**
 - Project UDI will update their specification as needed to provide real-time interfaces.
 - Project UDI does not plan to provide guidance as to how these interfaces are expected to be used or implemented in order to provide real-time behavior.
- **Issues not yet resolved**
 - We need to define the format and content of this specification or guide.



Issue Categories: Procurement Specification for Real-time

- UDI Operating System End users will be procuring operating systems that incorporate Real-time UDI. Project UDI specifies a UDI environment, not an operating system. Since the end user is not expected to procure a real-time UDI environment directly, there is a need for a way in which a user can specify the real-time capabilities that he needs when he orders his operating system. The user of this operating system should have a realistic expectation he will be getting.
- **Status Project**
 - UDI will update their specification as needed to provide real-time interfaces.
 - Project UDI specifications do not describe the characteristics of the operating systems that implement the UDI environment.
 - To specify real-time behavior, the characteristics of the operating system needs to be specified, not just the UDI environment.
 - R1.1 should develop a specification to fill this gap.
- **Issues not yet resolved**
 - We need to agree on whether this specification is needed and whether it should be a Project UDI specification or an R1 specification



Issue Categories: Certification for Real-time

- **UDI Operating System** To be able to procure a real-time UDI environment, there should be a way to certify. At the current time there is no test suite for UDI. Work has been started on a test suite that will probably be freely available. Its purpose is to aid vendors in developing quality drivers a a low cost. Its intent is not primarily for certification.
- **Status**
 - At the current time there is no test suite for UDI. Work has been started on a test suite that will probably be freely available.
 - Its purpose is to aid vendors in developing quality drivers a a low cost. Its intent is not primarily for certification.
 - Preliminary discussions indicate that Project UDI will be interested in technically supporting certification questions, but not specifying, a test suite, developing a test suit, or endorsing a test suite.
 - There may be more than one test suite and they do not want to unreasonably increase the cost of I/O driver developers.
 - The Open Group has indicated some interest in developing such a test suite.



Issue Categories: Certification for Real-time

- **Issues not yet resolved**
 - We need to agree on how to certify UDI for real-time and safety critical applications.
 - We also need to find the appropriate organization to develop the testing program and to do the certification.



Issue Categories: Distributed Time

- The DoD potentially has the need for a driver or protocol module to read a global (to multiple processors) real-time clock
 - Status
 - We need to define the requirements if any.
 - Issues not yet resolved
 - This subject has not yet been discussed.



Points of Contact

Russ Richards

Chair, NCITS TC R1, Real-time Computing Systems

Chief, Technical Architecture Division, Center for Standards
Joint Information Engineering Organization, Defense Information Systems Agency
10701 Parkridge Blvd.
Reston, VA 20191-4357

Phone: 703-735-3552

Fax: 703-735-3257

Office email: richar1r@ncr.disa.mil

Personal email: richardsrj@aol.com

Personal wireless: 703-898-8268

Bob Barned

Technical Editor, NCITS TC R1, Real-time Computing Systems

Senior Software Engineer

Lockhead Martin

EP7 RM333

PO Box 4840

Syracuse, NY 13221

Phone: 315-456-7101

Fax: 315-456-1430

Office email: robert.barned@lmco.com



National Committee for
Information Technology Standards

<http://www.ncits.org/tc_home/r1.htm>
NCITS TC R1, Real-time Computing Systems