Concurrent α

N91-1~~9726~~

320500

P.15

A New Generation of Real-Time DOS Technology for Mission-Oriented System Integration and Operation

E. Douglas Jensen

Concurrent Computer Corporation

Westford, MA

(508) 692-6200

edj@cs.cmu.edu, uunet.uu.net!masscomp!jensen

University of Houston RICIS and NASA/Johnson Space Center

Symposium on

Integrated Computing Environments for Large, Complex Systems

November 10, 1988

Outline

- System Integration and Operation (SIO) Requirements
- New Generation Technical Approaches for SIO

System Integration and Operation (SIO) OS Requirements

- Real-time
- Distribution
- Survivability
- Adaptability

SIO Application Requirements

Real-Time

- The application, and thus the OS, activities have various types of stringent *time constraints* (e.g., hard and soft deadlines) for their completion, which are part of the correctness criteria of the activities because they are critical to mission success and the survival of human life and property
- SIO is a dynamic and stochastic environment
    - a high percentage of the activities are aperiodic with critical time constraints
    - not all periodic and aperiodic time constraints can always be met, in which case application-specified recourse must be taken
- Activities have dynamic (time- and context-dependent) relative importance (functional criticality) as well as urgency (time criticality)
- The performance of the system, and of its OS, must be optimized for high-stress exception cases, such as emergencies (e.g., due to faults, errors, and failures, or even hostile attack)

SIO Application Requirements

Distribution

- Each system consists of many subsystems containing single- and multiple-processor machines which, for technical and logistical reasons, are loosely interconnected (i.e., via i/o paths such as buses or links) —

  in some systems, the subsystems may be physically dispersed across tens or even hundreds of meters

- These interconnected machines constitute a single integrated computing system, dedicated to a particular application, executing complex distributed programs

- A multiplicity of such systems communicate application data and status among one another, and are implicitly or explicitly coordinated in their mission activities —

  the distances among systems may be hundreds of meters

- System integration and operation is automated, and under the control of a (human) hierarchical command authority

SIO Application Requirements

Survivability

- The computing system must tolerate conditions far more severe than those encountered in non-real-time contexts
    - some systems are subject to hostile attack, so their hardware faults tend to be clustered in space and time
    - different systems have a wide variety of mission periods for which there is no single robustness approach: from hours to decades
    - limited or no repairs may be possible during the mission
    - the system usually has to remain in non-stop service during recovery from faults
    - extreme safety concerns: system failure may jeopardize the mission, human life, and property
- Because the hardware and software are distributed, there are multiple independent fault modes
- Overloads, faults, and resource contention are inevitably dynamic and stochastic
- Optimal performance under exceptional stress is the *raison d'etre* of the system

SIO Application Requirements

Adaptability

- Application limitations often demand maximum computing capability for the allowable size, weight, and power, which argues for special-purpose hardware and OS;

  but there is not just one set of fixed computing requirements

- There are many widely divergent real-time SIO applications, and the high costs of developing their computing systems argues for generality, standardization, and re-usability of the hardware and OS

- The computing requirements for any particular application evolve continuously over the entire lifetime of the system because
  - the application is extremely complex and difficult to understand
  - the application environment varies with time
  - technology advances rapidly

  and the application system lifetime can be decades

New Generation Technical Approaches for System Integration and Operation

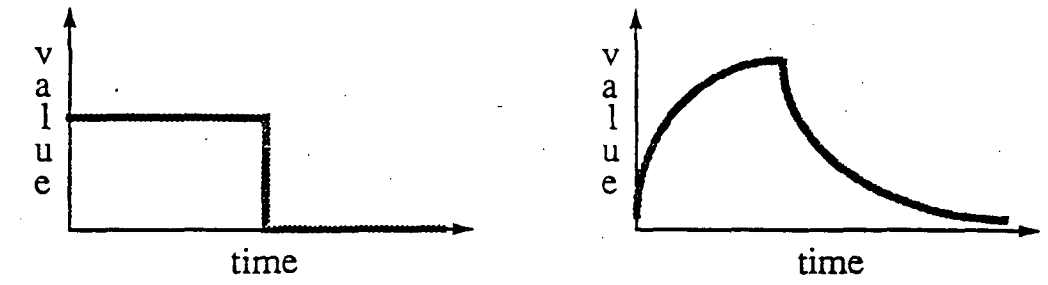
- Real-Time
- Distribution
- Survivability
- Adaptability

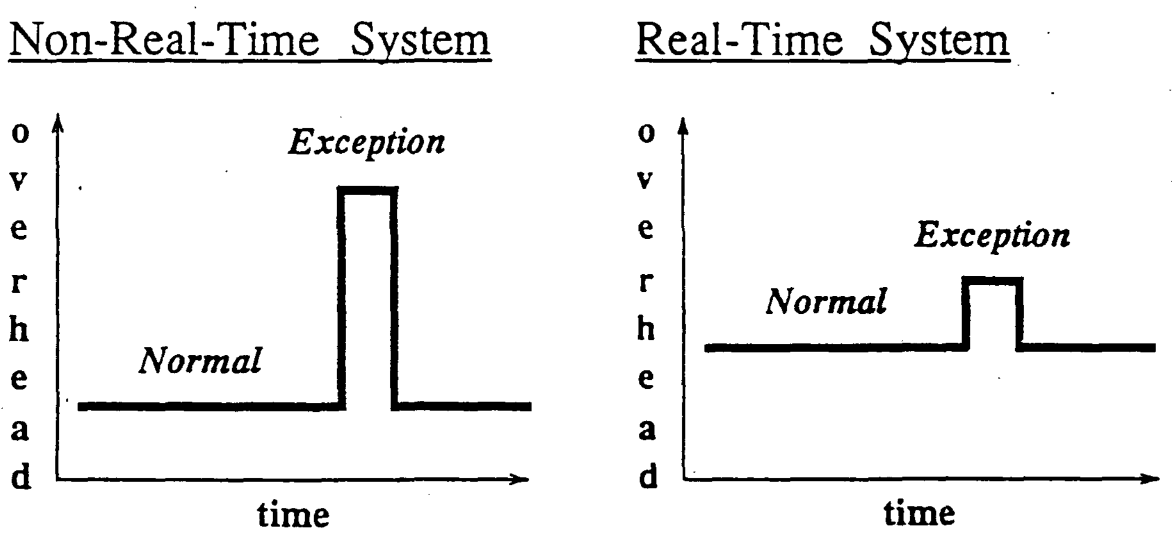
New Generation Technical Approaches for System Integration and Operation

Real-Time

- Manage all physical and logical resources directly with actual application-specified time constraints as expressed by *time-value functions* for all activities —
  - manages periodic and aperiodic activities in an integrated, uniform manner
  - distinguishes between urgency and importance
  - allows not only hard deadlines but also a wide variety of soft (i.e., residual value) time constraints
  - accommodates dynamic variability and evolution of both periodic and aperiodic time constraints
  - provides behavior which is as deterministic as desired and affordable
  - handles overloads gracefully according to application-specified policies
  - supports the clean-up of computations which fail to satisfy their time constraints, to avoid wasting resources and executing improperly timed actions
  - employs the same block-structured, nested, atomic commit/abort mechanisms as for transactions
- Optimize performance for exception cases

Sample Time-Value Functions





New Generation Technical Approaches for System Integration and Operation

Distribution

- Provide a new programming model which is well-suited for writing large-scale, complex real-time distributed software:

  objects (passive abstract data types — code plus data), in which there may be any number of concurrent control points; and *threads* (loci of control point execution) which move among objects via *operation invocation*

- A thread is a distributed computation which transparently (and reliably) spans physical nodes, carrying its local state and attributes for timeliness, robustness, etc.;

  these attributes are used at each node to perform resource management on a system-wide basis in the best interests (i.e., to meet the time constraints) of the whole distributed application

- Distributed computations must explicitly maintain consistency of data and correctness of actions, despite asynchronous real concurrency (and multiple independent hardware faults) — to accomplish this requires (at the kernel level, because the OS must itself be distributed)
  - real-time transaction mechanisms for atomicity, application-specific concurrency control, and permanence
  - system- and user-supplied commit and abort handlers

New Generation Technical Approaches for System Integration and Operation

Survivability

- The survivability properties and approaches include
  - graceful degradation: *best-effort* resource management policies; dynamic reconfiguration of objects
  - fault containment: data encapsulation (objects); object instances in private address spaces; capabilities
  - consistency of data, correctness of actions: concurrency control objects; resource tracking; thread maintenance; abort blocks; real-time transaction mechanisms (atomicity, concurrency control, permanence)
  - high availability of services and data: object replication; dynamic reconfiguration of objects
- The survivability features are presented through the programming model as a set of mechanisms which can be selected and combined as desired — their cost is proportional to their power
- Transactions
  - are scheduled according to the same real-time policies as are all other resources
  - allow application-specific commit and abort handlers

New Generation Technical Approaches for System Integration and Operation

Adaptability

- Adhere to the philosophy of *policy/mechanism separation*:
  - have a kernel of primitive mechanisms from which everything else is constructed according to a wide possible range of application-specific policies to meet particular functionality, performance, and cost objectives
  - provide these mechanisms at the optimal level of functionality — i.e., both necessary and sufficient to create large scale, complex real-time distributed systems
- Encourage application-specific information to be exploited statically and dynamically — e.g.,
  - special-purpose objects can be migrated into the kernel
  - references to objects can be monitored for locality
  - any attributes can be carried along with threads
  - special hardware augmentations can be objects
  - concurrency control and abort handlers can be special
  - resource management policies are application-defined
- Employ elastic resource management which flexes to tolerate variability in loading, timing, etc.

Alpha Program Management Overview

- Alpha originated at CMU-CSD as part of the Archons Project on real-time distributed computer architectures and operating systems—Doug Jensen was the Principal Investigator

- As part of a long-continuing "Think—Do" cycle, new concepts and techniques were created, based on the PI's 15 years of industrial R&D experience with real-time computer systems,

  then many of these were embodied in a feasibility test vehicle: the Alpha real-time decentralized OS

- The Alpha prototype ("Release 1")
  - lead by Duane Northcutt, with a team of five programmers for about three years
  - written for (homebrew) multiprocessor Sun workstations connected via Ethernet
  - consists of a high-functionality kernel, some system-layer functions, some software development tools
  - installed at General Dynamics/Ft. Worth in 1987 and demonstrated to many DoD agencies with a real-time C^2 application
  - numerous technical reports now becoming available

Alpha Program Management Overview
(continued)

- Alpha Release 2
  - intended to make the technology externally accessible, on reproduceable hardware platform, and further develop it
  - kernel interface spec subcontracted by CMU to Kendall Square Research, which Jensen later joined

    substantial functional enhancements were included
  - initial detailed design subcontracted to Concurrent when Jensen moved there
  - continuing research and remainder of design and implementation is part of a pending procurement
  - Jensen's Ph.D. students continuing research at CMU
  - pre-release available mid-CY89, release at end of CY89
  - portable, open, multi-vendor hosted
- Release 3
  - significant enhancements over Release 2
  - release at end of CY90