Volume : 3 | Issue : 1 | Jan 2014 • ISSN No 2277 - 8160

Research Paper | Computer Science

Real time system for software Engineering: An overview

Rajeeb Lochan Panigrahi — Asst. Professor, Department of Computer Sc & Engg, Gandhi Institute of Engineering and Technology, Gunupur-765022, Dist: Rayagada, Odisha

Murali Krishna Senapaty — Asst. Professor, Department of Information Technology, Gandhi Institute of Engineering and Technology, Gunupur-765022, Dist: Rayagada, Odisha

ABSTRACT

The purpose of this article is to provide a general introduction to computer control and to embedded computer systems. It defines what is meant by real time systems and show how real time systems can be classified. Further it illustrates the difficulties of writing real time soft-ware.

KEYWORDS : *Real time System*, System on a Chip (SOC), Smart Devices and MEMS Sensors, COTS Components, Fault Tolerant Systems, Real time UML

Introduction

The Real-Time software system presents an overview of time as it concerns technology convoluted schemes. Any scheme that responds at the pace of applicable events has real-time constraints whether the timescale is short, like the flight controls for an airplane, or longer, like the air travel reservation scheme for an airline. The aim is on solution of software to real-time problems¿answers that should be both correct and timely. programs development is examined with focus on real-time systems throughout each stage of the programs lifecycle. Real-time obligations investigation, architecting real-time systems, conceiving and modeling system timing, and implementation and checking schemes are studied.

Modeling methods utilising UML 2.0 are directed. Specific focus is placed on real-time arranging to accomplish yearned timing, reliability, and robustness. dialects and functioning schemes for real-time computing, and real-time troubles in concurrent and circulated schemes are discovered. This course presents a comprehensive outlook of real-time schemes with idea, techniques and procedures for the practitioner. The aim of this course is to inspire and arrange students to pursue more in-depth study of exact troubles in real-time computing and schemes development.

You are viewing the content of this page in Preview mode
Sign in or Register for full trial to access full content of this book

What is Real time system?

A real-time system is must process information and produce a response within a specified time, else risk severe consequences, including failure. It is, in a system with a real-time constraint it is no good to have the correct action or the correct answer ***after*** a certain deadline: it is either by the deadline or it is useless.

Define Real time system

Real- time processing normally requires both parallel activities and fast response. In fact, the term 'real –time' is often used synonymously with 'multi – tasking' or 'multi- threading'. Although there is no clear dividing line between real-time and non-real-time Systems, there are a set of distinguishing features

In the opinion from The Oxford Dictionary of Computing " Any system in which the time at which output is produced is significant. This is usually because the input corresponds to some event in the physical world, and the output has to relate to that same event. The lag from input time to output time must be sufficiently small for acceptable timeliness "

Young 1982 derived Any information processing activity or system which has to respond to externally generated input stimuli within a finite and specified period.

Type of Real time system

The real time system is one for which quantitative expressions of time are needed to describe its behavior. This quantitative expression of time usually appears in the form of a constraint on the time at which the task produces result. The most frequently occurring timing constraint is a deadline constraint which is used to express that a task is required to compute its results with in some deadline. We therefore implicitly assume only deadline type of timing constraints on tasks in this section, though other types of constraints as explained in sec 1.7 may occur in practice, real time tasks can be classified in to the following broad categories.

- Soft-Real Time System
- Hard Real Time System

Soft –Real Time System

Soft real-time systems the goal becomes meeting a certain subset of deadlines in order to optimize some application specific criteria. The particular criteria optimized depends on the application, but some typical examples include maximizing the number of deadlines met, minimizing the lateness of tasks and maximizing the number of high priority tasks meeting their deadlines.

Soft real-time systems are typically used where there is some issue of concurrent access and the need to keep a number of connected systems up to date with changing situations; for example software that maintains and updates the flight plans for commercial airliners. The flight plans must be kept reasonably current but can operate to a latency of seconds. Live audio-video systems are also usually soft real-time; violation of constraints results in degraded quality, but the system can continue to operate

Hard Real Time System

It is a system which is constrained to produce its results what in certain predefined time bounds. The system is considered to have failed when ever any of its hard real time tasks foes not produce its required results before the specified time bound. An example of a system having hard rasl time tasks is a robot.

The robot cyclically carries out a number of activities including communication with the host system, logging all completed activities, sensing the environment ot detect any obstacles present, tracking the objects of interest, path planning , effecting next move, etc. Now, consider that the robot suddenly encounters an obstacle. The robot must dectect it ad sa soon as possible try to escape colliding with it.If it fails to respond to it quickly (i. e, the concerned tasks ar not completed before the required time bound) then it would collide with the obstacle and the robot would be considered to have failure. Therefore, detecting obstacles and reacting to it are hard real-time task.

Another application having hard real time tasks is an anti-missile system that has to perform these critical activities (tasks), it must first detect all incoming missiles, properly position the antimissile gun, and then fire to destroy the incoming missile before the incoming missile

can do any damage. All these tasks are hard real-time in nature and the anti-missile system would be considered to have failed, if any of its tasks fails to complete before the corresponding deadlines.

Safety for critical system
In safety critical systems failure to respond in time is not an option. In the past some form of backup was provided to take over when the software system failed. But now systems are wholly software controlled real-time systems. Therefore we will concentrate on the design of hard real-time systems.

Current Technology
Current technology is making the design and implementation of Hard Real-Time Systems a lot more feasible.

We will give four examples.
System on a Chip (SOC)

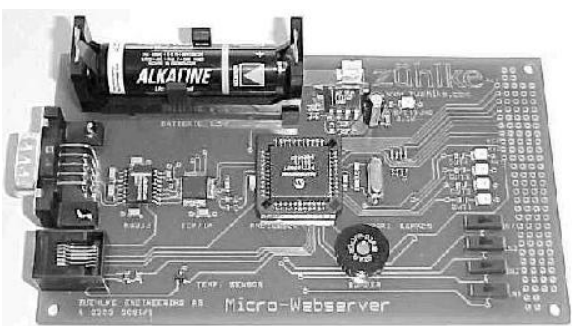
Nowadays it is possible to build an entire system on one chip including:

32 bit CPU

A megabyte of memory

I/O and network circuitry

They can be mass produced and thus have low production cost, although development costs tend to be quite high.



Webserver on a Chip

Smart Devices and MEMS Sensors
Smart devices are the combination of a sensor or actuator and a local microcontroller. Micro Electronic Mechanical Systems (MEMS) are sensor elements that can be integrated onto the same silicon die as the microcontroller.

Advantages of Smart Devices
Reduced interference from noise

Easier diagnostics

Plug and Play

Black Box

Easier inter connection

Cost Reduction

COTS Components
Reusing component made for earlier goods as an approach to new system development is a promising way of accomplishing the cited development and system improvements. There is furthermore the possibility to buy software components from concerned vendors, so called Commercial-Off- The-Shelf, COTS, and component. In the current development system, the use of Commercial-Off-The-Shelf, software component is drastically expanding. Using COTS Components can be one way of reducing development time and be comparable by getting goods to the market very quick and inexpensively. COTS constituents can also supply an advanced reliability compared to custom-made components since they are refined by considerable field-testing. Certain types of constituents are rarely evolved solely for one intended system anymore.

Type of COTS Components
There are three main types of COTS Components
- Hardware components – e.g. clocks
- Software Components – NB no certain temporal properties.
- Hardware/Software components – e.g. Smart sensor.

Fault Tolerant Systems
Systems that will continue to function, even when parts of the system fail. Achieved by replicating critical functionality and providing diagnostics about the state of the system For i.e: Airbus A340 Flight Control Software

Construction of Hard Real Time Systems
From the current technology trends it seems as if the future lies in distributed Real-Time

Systems. They will consist of a network of nodes, these nodes will either be: Powerful system nodes (SOC's), or Smart sensor nodes, and

Top Down or Bottom Up.
This Real-Time Systems can be designed top down or bottom-up. Top down focuses on the architecture where as Bottom up on the individual components.

Composability
As the system is a distributed system it will consist of different components. The communication network interfaces (CNI's) between these components need to be well defined in both the value and temporal domains

An Ideal Component
Ideal component is a unit of service provision, Validation, error containment and a unit of design of maintenance. Considering these factors a hardware/software component seems the best choice for a real-time system.

Principles of Compos ability
For a component to be integrated into a real-time system it must fulfill the principles of composability.

1) Independent development
2) Stability of prior services
3) Constructive Integration

Validation
Validation is a process of ensuring that the data that are entered fall within the accepted boundaries of the application collecting the data. The validation can be classified below categories.

Product Validation - Must be able to validate components independently of the system context.

Worst Case Execution Time – Establishing upper bound on worst performance.

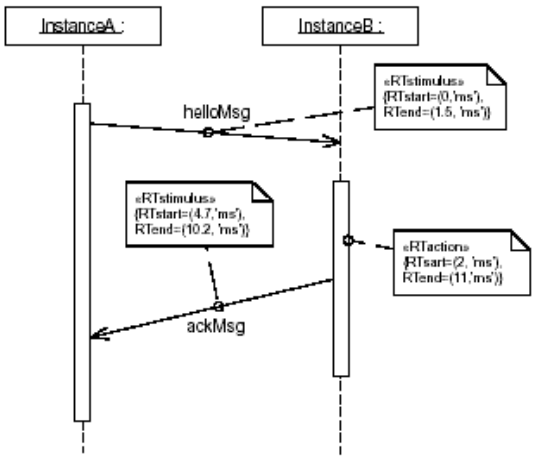
Simulation – Generates events which stress the system.

Formal Verification – verifies underlying algorithm.

Real-Time UML
UML combines graphical and textual notations to describe systems such as this. UML is primarily a set of notations and does not prescribe a specific development process. An extension to UML is providing support for developing Real-Time Systems. It also provides techniques and notation for modeling and analyzing Real-Time Systems. Not currently rigorous enough to allow temporal interface specification.

Sequence Diagram
Sequence Diagrams provide a logical place in the modeling view to capture this timing information. The timing requirements associated with a particular system function are modeled in the Use Cases and mapped through the Sequence Diagram(s) onto specific interactions between objects. Where UML defines the notation and diagram con-

tents, modern modeling tools supporting .



Sequence Diagram

Conclusion

In this paper we have looked at the problem of real-time software , concentrating on two basic aspects of real time system and real time software. Finally, we presented some of the technologies and the tools that are available to real-time software developers. As society moves in the direction of a symbiotic dependency on computer systems, more and more of these systems will fall in the broad class of real-time systems. This means that the methods evolved for traditional real-time software, encompassing in particular methods for fault tolerance and predictability will become part of the benchmark knowledge groundwork of most software engineers and developers. Furthermore, they will be supported by progressively more sophisticated

Real-Time Software Systems are becoming more viable due to technology advances. They require different design and verification techniques to non Real-Time Systems. The techniques that currently exist aren't good enough. When they are there is a very bright future for Real-Time Software Systems.

REFERENCES

| Burns, A. and Wellings, A.,(1997), "Real-Time Systems and Programming Languages," (2nd ed.) Addison-Wesley. | Mal, Rajib, (2011) Real time system : theory and practice , Pearson Education, New Delhi | M. Callieri, P, et all.,(2006), A realtime immersive application with realistic lighting: The Parthenon, Computer Graphics, 30,368-376. | Jinfang Zhang, Jon W. Mark, Xuemin (Sherman) Shen, (2005), An adaptive handoff priority scheme for wireless MC-CDMA cellular networks supporting realtime multimedia applications, Computer Communication,25,1240-1250. | Klein, M., et al.,(1993) "A Practitioner's Handbook for Real-Time Analysis: Guide to Rate Monotonic Analysis for Real-Time Systems,"Kluwer Academic Publishers. | Selic, B., Gullekson, G., and Ward, P.(1994) , "Real-Time Object-Oriented Modeling,", John Wiley & Sons. | Saksena, Manas Bran Selic,(1999) " Real-Time Software Design-State of the Art and Future Challenges" Canadian Review. | Crnkovic and M. Larsson (2002), "Building Reliable Component-Based Systems", Artech House, ISBN 1-58053-327-2