A Review of the Scopes and Challenges of the Modern Real-Time Operating Systems

Amit K. Shukla, Department of Computer Science, South Asian University, New Delhi, India
Rachit Sharma, Department of Computer Science, South Asian University, New Delhi, India
Pranab K. Muhuri, Department of Computer Science, South Asian University, New Delhi, India

ABSTRACT

A real-time operating system (RTOS) is an integral part of a real-time embedded system (RTES). Most of the RTESs work on dynamic environments, and hence, the computational cost of tasks cannot be calculated in advance. Thus, RTOSs play a significant role in the smooth operations of the RTES through efficient task scheduling schemes and resource managements. This article investigates the existing design challenges and scope of the modern RTOSs. A wide variety of latest RTOSs are discussed and elaborated in detail. A comparative study with their prospects has been explained so that interested readers can use the article as a readily available starting point for their further studies on this topic.

KEYWORDS
License, Programming Language, Real-Time Operating Systems, Real-Time Systems, Scheduling

1. INTRODUCTION

Real-Time Operating Systems (RTOSs) are a special class of operating systems (OS), used to serve Real-Time Embedded Systems (RTESs) or a Real-Time Systems (RTSs). RTOSs help RTESs to operate in the presence of numerous constraints such as expended power, response time, and maintaining an efficient usage of memory. RTOSs are used in embedded computing systems for resource allocation and planning to carry out multiple tasks within worst case execution times pertaining to a particular architecture. The major concern with RTESs is that they should produce an accurate output within a deadline or an allotted time frame. The RTOSs are crafted scalable enough for various sets of applications for different RTESs.

RTESs are categorized into three classes, viz., hard, soft, and firm RTES. Graphical representation of hard, soft and firm RTES are presented in Figure 1.

A hard RTES is prone to less jitter than its soft counterpart (Mall, 2009). A RTES is classified as a soft RTES, if the output of task remains useful even if it’s execution requires violation of the deadline up to a certain extent. However, if in a system, the violation of a task deadline is associated with the possibility of huge damage or catastrophe, then it is termed as a hard RTES. When, output of task loses its utility beyond the task deadline, then then system is called a farm RTS.

DOI: 10.4018/IJERTCS.2018010104

Copyright © 2018, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.
Figure 1. Graphical representation of RTOS

![Graphical representation of RTOS](image)

(a) Soft RTES | (b) Hard RTES | (c) Firm RTES

Systems employed in avionics controllers find extensive application of RTOS. Therein, a single computer is used to control several aircraft subsystems mandating an OS with temporal and spatial partitioning systems. Spatial partitioning is a concept that refers to the function of isolating tasks in the computer memory, and temporal partitioning is concerned with efficient utilization of processor time. These partitioning schemes allow a single processor to run several tasks in parallel at a particular instant of time. Therefore, we see that the role of RTOSs is immensely significant for the purposeful and secure operations of the RTESs.

There are various requirements in a RTOS such as predictability of task, strictness towards deadline violations, configurability of system, size, functionality, efficiency of system components i.e. time granularity, threads, and resource management etc. (Joe & Kim, 2017; Alimi et al., 2017; Dietrich et al., 2017; Guan et al., 2016). However, the primary objectives of RTOSs to utilize efficient scheduling techniques with good resource management policies, and predictable performance of tasks. Scheduling, weather hardware or software, is one the crucial factor for real-time tasks (Lin et al., 2015; Zhang et al., 2011; Gracioli, G., & Fröhlich, 2017). Moreover, an important part of the OS kernel is the scheduler which is driven by periodic clock interrupts, except during the occurrences of voluntary context switches. As shown in Figure 2, architecture of RTOS for complex embedded systems can be a combination of various modules including kernel, stacks, networking protocol, device drivers, and file system.

Figure 2. Architecture of RTOS

![Architecture of RTOS](image)
Quality of Services and Optimal Management of Cloud Centers with Different Arrival Modes
www.igi-global.com/chapter/quality-of-services-and-optimal-management-of-cloud-centers-with-different-arrival-modes/105679?camid=4v1a

Strategic Scenarios for Fixed-Mobile Convergence: An Integrated Operator Case
www.igi-global.com/chapter/strategic-scenarios-fixed-mobile-convergence/54009?camid=4v1a
Distributed Cooperative Synchronization for Large-Scale 4G Wireless Sensor Networks Using CAZAC Sequences
www.igi-global.com/article/distributed-cooperative-synchronization-large-scale/64228?camid=4v1a

Digital Convergence and Home Network Services in Korea: Part 2 – Business Models and Strategic Alliances
Hyun-Soo Han, Heesang Lee and Yeong-Wha Sawng (2009). *Handbook of Research on Telecommunications Planning and Management for Business* (pp. 234-246).
www.igi-global.com/chapter/digital-convergence-home-network-services/21668?camid=4v1a