Feature and Performance Analysis and Enhancements of the DECOS Core OS

Andreas Wolf[1], Maximilian Rosenblattl[1], and Bernhard Leiner[1]

TTTech Computertechnik AG
Schoenbrunner Strasse 7, 1040 Vienna, Austria,
phone: +43 1 5853434 0, fax: +43 1 5853434 90,
bernhard.leiner@tttech.com

Abstract. This paper presents an overview of the features of the Core Operating System (COS) of the Encapsulated Execution Environment (EEE) developed in DECOS (Dependable Embedded Components and Systems), an integrated project within the Sixth Framework Programme of the European Commission.
For all features, the current status of the tool support is shown, combined with how these features are used in current applications.
Additionally, this paper presents ideas to improve the performance of the COS by adapting the COS to the way current applications use it and by removing features currently not in use. Also, reducing the feature set simplifies the configuration and potentially enhances the reliability.

Keywords: Embedded Systems, Dependability, Virtualization.

1 Introduction

The Core Operating System (COS) of the Encapsulated Execution Environment (EEE) developed in DECOS (Dependable Embedded Components and Systems), an integrated project within the Sixth Framework Programme of the European Commission, has a rich feature set while providing a high level of protection for the encapsulated partitions. When designing the COS, the focus was on building an operating system that ensures protection of user partitions, mainly driven by reliability and safety considerations. However, some requirements specify huge flexibility of the execution environment at runtime by requiring reconfiguration of e.g. timers and external events.

Interestingly, dynamic reconfiguration is not used at all by the project partners and many operating system features are used very rarely because of the very low performance of system calls.

This paper presents the current feature set of the COS and gives reasons for the low performance of the COS operations. Possible enhancements are discussed and their possible impact on performance is analyzed.

It is shown that a strictly deterministic time-triggered approach has advantages against the current implementation regarding performance, determinism and reliability.