A Scenario-Based Approach for Predicting Timing Properties of Real-Time Applications

Egor Bondarev
Eindhoven University of Technology
P.O. Box 513, 5600 MB Eindhoven, Netherlands
Phone: +31 (0)40 247 2480
E-mail : E.Bondarev@tue.nl

Peter H.N. de With
LogicaCMG/TU Eindhoven
P.O. Box 513, 5600 MB Eindhoven, Netherlands
Phone: +31 (0)40 247 8210, +31 (0)40 29 57 777
E-mail : mailto:P.H.N.de.With@tue.nl

I. INTRODUCTION

Embedded systems are often characterized by two closely coupled properties: limited resources and real-time constraints for executing running applications. The limitation of resources, such as memory size, memory bus and processing power, makes it more difficult to guarantee the real-time execution of applications. However, having that guarantee is crucial for e.g. multimedia devices.

During the design phase, in order to ensure that an application will fit on a target device, it is important to determine or predict the resource usage of an application. The resource-usage *prediction* is a technique to estimate the amount of consumed resources by analyzing the design and/or implementation of an application.

In the Space4U[1] project [2], which is an extension of the ROBOCOP project [1], a component-based architectural framework was introduced for the middleware development of high-volume embedded devices. Component-based development complicates the resource-usage prediction per application, because actual resource consumption is distributed over individual components. In this paper, we propose a technique for predicting the timing property of a component composition, also called a component assembly.

The key problem of such a *predictable assembly* is to first find and express a component's timing property, and, second to combine them in order to make predictions over a composition of those components. It should be noticed that there is a clear difference between the component and application timing properties. In case of component development, the designer deals with metrics such as worst-case, mean-case and best-case execution times per function. In case of component composition or application development, the designer focuses on finding the following properties: end-to-end response time and processor utilization bound of an application.

II. COMPONENT DEPENDENCIES

In the Space4U project and in this paper, we propose a predictable assembly technique that allows the translation of an already known property of components into a property of an application assembling these components. This technique is completing and refining the coarse scenario-based predictable assembly model [3] that was proposed in the ROBOCOP project.

A primary benefit of the ROBOCOP framework is that a component designer specifies not only provided interfaces, but also required interfaces of a component. While provided interfaces help an application developer to find a component that would do the work, the required interfaces specify what other components a particular component may depend upon. Finally, the provided and required interfaces allow the application developer to describe explicitly static component dependencies via interfaces within an application (see the simple decoder example in Figure 1).

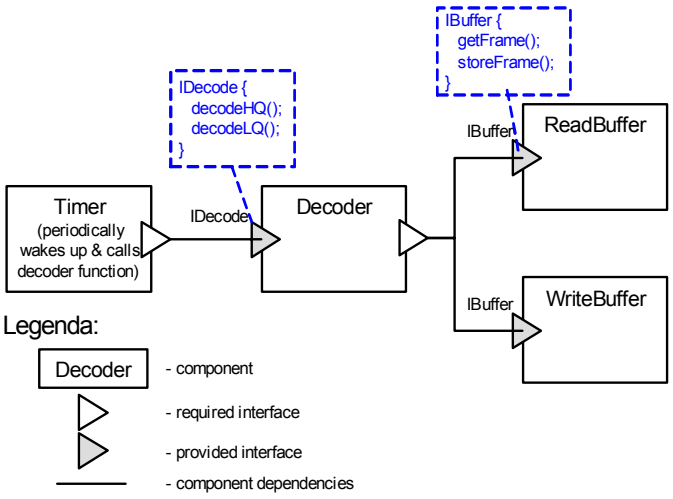

Figure 1. Static component dependencies for decoder

[1] Space4U is part of the ITEA research programme funded by the European Union.

Normally, each interface encapsulates a set of functions. Taking into account interface dependencies and, analyzing the most commonly used scenarios of an application, a developer can identify function call sequences per task. Following this, if the timing properties (WCET) of those functions involved in a task execution are given by a component developer, we can find the timing property of the complete task, while considering the commonly used scenario. The scenario can be represented by a sequence chart diagram (see the "high-quality decoding" scenario in Figure 2).

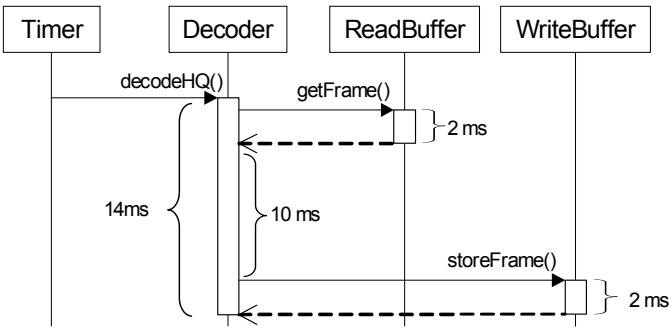

Figure 2. Sequence chart diagram for a decoding task

Thus, with the predictable assembly technique, we can derive the *execution (response) time* of a complete task in an application.

Unfortunately, the ROBOCOP predictable assembly model does not consider two important properties: using a multitude of tasks in an application and means for synchronization between tasks. Especially the last property is extremely important for real-time performance analysis.

III. An Improved Model For Prediction

In order to find a processor utilization rate of a task, the execution time can be divided by a period or minimum inter-arrival time of a task. The cumulative processor utilization rate of a set of application tasks represents the application *processor utilization bound*.

The described scenario-based approach in Section II is a cornerstone for the predictable assembly technique. To apply the technique in practice, we introduce an *assembly description language*. The language allows an application developer to specify all behavioral and aspects of an application that may influence resource usage. We have found that a significant part of those aspects deal with synchronization of tasks. The final result of the specification is an application task model.

The designer, while writing the specification, takes as an input (see Figure 3):

a) Available component description (incl. WCET per function and provides/requires interfaces),
b) Commonly used scenarios.

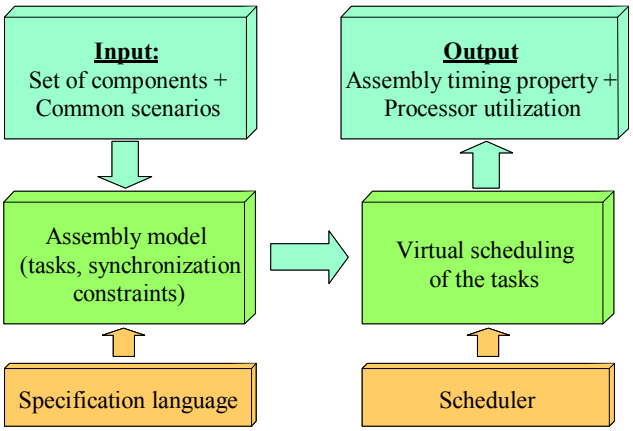

Figure 3. Work flow of the predictable assembly technique

In the assembly specification model, the designer specifies component dependencies, describes tasks and corresponding function sequence calls, and finds synchronization aspects between the tasks. When the specification is available, it actually represents application tasks properties. Having the tasks properties, it is possible to schedule these tasks. Therefore, the final step of the predictable assembly technique is to apply virtual scheduling on the application model. As a result, the following application timing properties are found:

- Response time of critical tasks,
- Processor utilization bound,
- Schedulability of the application on a target.

IV. Conclusions

The improved predictable assembly technique enables the prediction of application timing properties already at the design stage. The technique can be extended for predicting memory and bus usage.

For future work, we propose to conduct several case studies with multimedia applications and investigate a prediction-error rate of the technique for different application domains.

References

[1] ROBOCOP public home page [http://www.extra.research.philips.com/euprojects/robocop/]
[2] Space4U public home page [http://www.extra.research.philips.com/euprojects/space4u/]
[3] Merijn de Jonge, Johan Muskens and Michel Chaudron. *Scenario-Based Prediction of Run-time Resource Consumption in Component-Based Software Systems.* In Proceedings of 6th CBSE conference. May 3-4, 2003