Domain-Specific Programming Environment for Heterogeneous Multicore Embedded Systems

Alexey Syschikov, Saint Petersburg State University of Aerospace Instrumentation, Saint Petersburg, Russia
Yuriy Sheynin, Saint Petersburg State University of Aerospace Instrumentation, Saint Petersburg, Russia
Boris Sedov, Saint Petersburg State University of Aerospace Instrumentation, Saint Petersburg, Russia
Vera Ivanova, Saint Petersburg State University of Aerospace Instrumentation, Saint Petersburg, Russia

ABSTRACT

Nowadays embedded systems are used in a broad range of domains such as avionics, space, automotive, mobile, domestic appliances etc. Sophisticated software determines the quality of embedded systems and requires high-qualified experts for software development. Software becomes the main asset of embedded systems that is valuable to retain in changing computing platforms in embedded systems evolution. Computing platforms for embedded systems became multicore processors and SoC, they can change in the embedded system lifetime that could be long (dozen of years for an automobile and airplane). It requires software porting to new platforms as a regular process. Many tools and approaches allow developing of software for domain area experts, but mainly for general-purpose computing systems. In this paper the authors present the complex technology and tools that allows involving domain experts in software development for embedded systems. The proposed technology has various aspects and abilities that can be used to build verifiable and portable software for a wide range of embedded platforms.

Keywords: Domain-Specific Language, Embedded Software, Embedded Systems, Portable Software, Programming Environment, Visual Programming

1. INTRODUCTION

The continuous miniaturization of computing devices has contributed to making embedded systems a wide variety of diverse computational requirements (Kuhn T., 2009). Regardless of the device, be it mobile phone, vehicle equipment, medical instrument, or smart home component, all of these systems embody very stringent requirements in terms of reliability, maintainability, availability, safety, security, efficiency, energy consumption, among others. Overall, the diversity

DOI: 10.4018/IJERTCS.2014100101
of embedded systems and requirements pose tremendous challenges to the development and robustness of their software applications. Transition to multicore/many core embedded computing platforms make the problem especially tough. In particular, this software must operate with acceptable performance parameters in resource-constrained environments while being subject to changing operating conditions (e.g., temporary unavailability of sensors, decreasing battery level, real-time requirements, memory limitations, intermittent connectivity).

A software developer for embedded systems faces the set of challenges, which raise over a traditional systems programming. Embedded systems grow from simple control systems to advanced intelligent systems. Embedded systems are domain-oriented both in software and hardware aspects. Embedded development results have strict requirements and constraints to non-functional characteristics.

Such kind of challenges cannot be effectively solved with traditional set of programming instruments. New methods and tools should reflect the specifics of embedded systems, aspects of embedded software development and software-hardware interrelations.

2. THE PROBLEM STATEMENT: FROM DOMAIN TO MULTICORE PLATFORMS

In contrast to general-purpose computing systems, most embedded systems are domain-oriented. This means that in embedded world a single system solves just one or several tasks within the single application domain.

Early embedded systems were limited in hardware and thus solve quite observable tasks, mainly in the control domain. The primary skill for software developer was a programmer’s skill, and there was no need in deep domain and task understanding.

Nowadays the complexity of tasks running on embedded systems is increasing rapidly due to rapid growth of hardware performance. There is a transition from simple control systems to high-performance systems for signal, image, video processing (Joachim Keinert, Jürgen Teich, 2011; Sedcole N. P., Cheung P. Y. K., & Luk W., 2003) and even for decision making (Song Z1, Ji Z, & Faust O, 2011). An algorithmic and a programming workload of such tasks also increase fast. Embedded system programmers are no longer able solving such complex tasks alone. It should be a “two-in-one” developer: skilled domain expert, who can develop algorithms for task solution, and a skilled programmer, who can implement such algorithm efficiently. For heterogeneous multicore embedded platforms, such developer additionally should have a deep understanding in parallel programming and in programming for specific heterogeneous systems.

If we look on all stages of embedded software development (Figure 1), we can see that there are a set of gaps in interaction between domain experts, programmers and hardware experts.

We need to go deeper in details of these gaps and the ways to fill them.

2.1. Programmability: From Domain to Program

The “All-in-one” specialists are too rear to fulfill all requirements in an embedded software development. Thus, we need to separate domain experts and qualified programmers and involve both of them in the process of embedded software development.

Here we faces the gap between a domain expert, who describe the task algorithm in domain terms and in a “paper mode” (documents, algorithm schemes, time diagrams, etc.), and a programmer, who implement the task solution structure and extend it with exact data processing code. This leads to multiplication of required effort and to potential misunderstanding between
Theoretical Analysis and Experimental Study: Monitoring Data Privacy in Smartphone Communications
[www.igi-global.com/article/theoretical-analysis-and-experimental-study/79282?camid=4v1a](www.igi-global.com/article/theoretical-analysis-and-experimental-study/79282?camid=4v1a)

Cross-Layer Design for Network Security Enhancement
[www.igi-global.com/chapter/cross-layer-design-network-security/65668?camid=4v1a](www.igi-global.com/chapter/cross-layer-design-network-security/65668?camid=4v1a)