SPECIAL ISSUE SECTION

Dependability and Safety of Real-Time Computer Systems

Editors:

Wojciech Grega,
Andrew J. Kornecki,
Janusz Zalewski

Editorial

Special Issue on Dependability and Safety of Real-Time Computer Systems

It is our great pleasure to present our readers with this special issue of JAMRIS on *Dependability and Safety of Real-Time Computer Systems*. The articles in this issue are extended and revised versions of papers that have been selected from submissions to the RTS'07, 2nd Workshop on Real-Time Software, which took place as a part of the IMCSIT Conference (International Multi-conference on Computer Science and Information Technology), organised by the Polish Information Processing Society (PTI), in Wisła, Poland, 15th-17th October 2007 (please see http://www.imcsit.org/ for details about this series of conferences).

In today's society, computers are omnipresent and are associated with almost all our activities. Computers are used in control and monitoring not only in industrial and scientific applications, as it used to be a few years ago, but also in everyday life. Digital, microprocessor-based devices are present in every aspect of our daily life, from door locks and alarm clocks, to cell phones, to cars, traffic lights, and medical equipment, banking terminals, airplanes and space probes. The common characteristic of all these computer-controlled devices and systems is that they depend on the proper operation of microprocessors that are embedded in them, and run in real time, that is, their response time is bounded.

As a consequence, the widespread use of computers, which are all controlled by software, may pose significant problems related to software dependability and safety. *Dependability* is a term in common use, but the accepted definition is not that well known. Dependability is "the property of a computer system, such that reliance can be justifiably placed on the services it delivers" (IFIP WG10.4, 1989). Such definition reflects the fact that the concept of dependability is a complex one, and has several aspects. One of these aspects involves dependability attributes, i.e., critical properties that constitute the notion of dependability. Among such properties, those most commonly listed are: reliability, safety, and security.

Reliability and security are traditional properties of computer systems. Reliability is defined as "performing required functions under stated conditions for a specified period of time" (IEEE Glossary of Software Engineering Terminology). The security essentially deals with protection from external threats. However, computer or software safety is a term not clearly defined and often misinterpreted. Essentially, it can be described as freedom of risk, to a human or a society, caused by a computer or software. In this sense, safety is a property very different from reliability, which reflects only performing required functions without taking into account risks, and is a direct inversion of security, in a sense that security deals with protecting a computer system from external threats, while safety deals with protecting the external world from the consequences of a computer malfunction.

Computer systems that require special consideration regarding safety, for example in applications such as flight control and traffic control, road vehicles, railway interchanges, nuclear facilities, medical equipment and implanted devices, etc., are called safety-critical systems. The question fundamental to the development of safety-critical computer systems is: *How to achieve and improve dependability and safety?*

Taking into account the typical development cycle of such systems, composed in the simplest case of three general stages: requirements specification, design, and implementation, one can look at ways of addressing dependability and safety at each individual stage. This is, roughly speaking, how the papers in this issue are organized.

At the requirements specification stage, the dominating research approach is the use of formal methods, i.e. rigorous mathematical models and techniques for the specification of system properties. Three papers in this issue represent such research paradigm. Leuchter, Tyszberowicz and Feldman discuss a method and tool to translate specifications expressed in *Verilog* into an equivalent representation in a synchronous language

Esterel. Their tool, *Veriest*, is a translator that converts respective designs. Since many libraries of useful designs, such as communication protocols, compression algorithms, are available in Verilog, the large body of intellectual-property hardware designs is thus available to be incorporated into a synchronous language.

The second paper of this category, by Olga Tveretina, discusses verification of real-time systems with the *Coq* proof assistant. She deals with a hybrid automaton as a mathematical formalism for describing hybrid systems, that is, systems involving the interaction of discrete and continuous dynamics. A framework for reachability analysis is presented, which allows avoiding spurious transactions for certain classes of hybrid systems. Kosęda, Szmuc and Cichalewski use a formal approach in designing automation software for the free electron laser. They present a method for formal specification and verification of software, based on model checking with *NuSMV* tool. The tool is used to verify formal properties included in the specification of software. A dedicated converter translates the model expressed in the specification language into the NuSMV's input language. Definitions of formal properties to be fulfilled by the model are expressed in Computational Tree Logic (CTL).

Formal notations work well only for perfect mathematical models, which are rarely achievable in practice. So, alternative engineering techniques are needed at the design level for more practical solutions. Two papers addressing such approach are included in this issue, both based on developing more realistic architectural models, using the *UML* modelling language, specifically the statecharts, to express designs related to improving safety. Letia, Barbu and Dinga propose to increase software dependability directly addressing the safety issues in road traffic control, by developing a general pre-emptive scheduling algorithm. It is based on using local priorities to schedule critical resources and global priorities to impact global traffic patterns. The simulation results demonstrate that the proposed solution provides robust performance for relatively low traffic volumes.

The second paper in this category, by Lu and Halang, deals with combining component-based software architecture models with established fault tolerance techniques. Fault tolerance is often used in research and practice as a method of improving dependability and safety. It is defined (IEEE Glossary of Software Engineering Terminology) as “the ability of a system or component to continue normal operation despite the presence of hardware or software faults”. As such, it encompasses special techniques to mitigate or avoid consequences of faults. The paper presents an architecture described in terms of normal- and abnormal-activity components aiming to support a wide variety of fault tolerance features, and suggest extending UML to express error detection, error recovery, and redundancy measures to help improve dependability.

The third category of papers involves experiment-based analytical approach to dependability and safety and is particularly effective at the implementation level. Three papers are included in this category. Gawkowski and his colleagues address the issue of fault sensitivity in two specific classes of control algorithms: Dynamic Matrix Control (DMC) and Generalized Predictive Control (GPC). Dependability of both algorithms is evaluated using a software-implemented fault injection technique. Tests performed on the control system of a remotely controlled robot vehicle show that the considered algorithms may exhibit unacceptable output response, such as slow or oscillatory behaviour, and require exception handling to address these and related problems.

Piątek and Grega also analyse the behaviour of a digitally implemented controller for a magnetic levitation device, by studying its speed. Their concern is that the performance of a digital control system depends not only on variables, such as the sampling period, control loop execution time, jitter, and general performance of individual components, but also on their interaction and cooperation. They propose a new design approach based on the relative speed system classification that relies on balancing the closed-loop execution time and process dynamics. The detailed analysis and optimisation of control algorithm performance usually involves dealing with the implementation at the operating system level. This is the focus of the last, but not least, paper in this category. Moryc and Èernohorský present an analysis of real-time task jitter under *RTLinux* operating system. They describe methods and tools developed to measure jitter, and discuss the results obtained for a data acquisition system. They conclude that the Linux kernel itself significantly contributes to the real-time task jitter, and suggest that employing a CPU designed specifically for real-time operation could mitigate the effects.

Discussion of dependability and safety of real-time computer systems would not be complete without touching one extremely important subject: education. Therefore, we have included a paper on curriculum development for real-time software-intensive control systems, by Kornecki *et al*. The authors report on the international activities, involving American, Polish, Czech and French universities, focused on designing and implementing a coordinated curriculum that would engage students from multilingual, geographically

separated institutions, and expose them to problems, methods, solution techniques, infrastructure, technologies and tools in the domain of dependable real-time safety-critical control systems. The hope is that joint efforts will help in creation of international curricula with a compatible quality assurance and assessment process, enabling the mobility of the future workforce and facilitating the career advancement.

We hope that this snapshot of problems and solutions important to dependability and safety of real-time computer systems, collected in a single issue, will be of interest to the readers of JAMRIS. Those interested in this subject more deeply may consider submitting papers to the next edition of the RTS Workshop, which is planned for the 2009 IMCSIT conference in Mrągowo, on 12-14 October 2009. For details, please see http://www.imcsit.org/ or contact the editors of this special issue.

Wojciech Grega
AGH University of Science and Technology, Kraków, Poland
wgr@ia.agh.edu.pl
Andrew J. Kornecki
Embry-Riddle Aeronautical University, Daytona Beach, Florida, USA
kornecka@erau.edu
Janusz Zalewski
Florida Gulf Coast University, Fort Myers, Florida, USA
zalewski@fgcu.edu