Performance Estimation in complex Automation Systems

Matthias Pfragner
honk@sbox.tugraz.at
Institute for Technical Informatics
TU Graz, AUSTRIA

Peter Priller
peter.priller@avl.com
AVL List GmbH
Graz, AUSTRIA

Dietmar Prisching
dietmar.prisching@avl.com
AVL List GmbH
Graz, AUSTRIA

Bernhard Rinner
rinner@iti.tu-graz.ac.at
Institute for Technical Informatics
TU Graz, AUSTRIA

Abstract – Automation systems (AS) are important tools for research and development. They are connected to the physical system under test via numerous sensors and actuators. Due to the ever increasing requirements in performance and functionality automated support in the configuration of the automation system's hardware and software components is desired. A prerequisite for automated configuration is the prediction of the performance of the automation system.

At the previous TCMC workshop we have presented our approach for modeling and predicting the performance of our automation system. In this paper we extend this approach to be able to predict the performance of both real-time and non real-time processing on the automation system. We further present recent results from modeling and prediction of complex configurations of the PUMA Open automation system and compare the predicted parameters with the measured performance values. The PUMA Open automation system is targeted for the design and test of engines, transmissions and power trains.

Keywords: automation systems, performance estimation, response time analysis

I. INTRODUCTION

Automation systems (AS) are nowadays essential tools for research and development especially in the automotive domain. Functionality and complexity of such tools have been significantly raised over the last years in order to fulfill the ever increasing requirements. A typical AS is connected to the physical system under test via numerous sensors and actuators and automatically performs various measurements and test procedures. The most challenging requirements for such an AS are, therefore, the integration of various hardware and software components, the real-time data processing and the configuration of the overall system. Due to the complexity of modern AS support in the configuration of the hardware and software components is required. The determination of performance characteristics is an important precondition for a configuration support.

Most complex automation systems are realized as a combination of real-time (RT) and non-real-time (NRT) systems. The real-time part is responsible for data acquisition, safety and control functionality. On the other hand, the non-real-time part provides data processing, data persistence and the user interface. Due to the ever increasing processing power of current computers the complete automation system may be integrated on a single computer platform. In this case RT as well as NRT operating systems are installed together on the single processor platform.

Figure 1: The PUMA Open Instrumentation and Test System for Engines, Transmissions and Power Trains

In [1] we presented our approach for modeling and predicting the performance of complex automation systems. This approach has been focused on predicting the timing behavior of all real-time tasks by response time analysis [2,3]. In this paper, we extend this approach to be able to predict the performance of both real-time and non real-time processing on the automation system. We further present recent results from modeling and prediction of complex configuration of the PUMA Open automation system and compare the predicted parameters with the measured performance values.

The remainder of this paper is organized as follows: Section 2 introduces the PUMA Open AS which is used as case study. Section 3 presents related work as well as our performance modeling and prediction approach. Section 4 describes the current status and experimental results. Section 5 concludes this paper with a brief discussion.

II. PUMA OPEN AUTOMATION SYSTEM

PUMA Open is an automation system (AS) (Figure 1) for the development and test of engines, transmissions and power trains. PUMA Open has been designed as an open platform in the sense that it is based on standardized interfaces for data acquisition and communication as well as modular hardware and software components. This supports the extension and configuration of the AS.

Figure 2 presents a part of the PUMA Open instrumentation interface. It supports various bus systems such as IEEE1394, CAN, Profibus, RS232, T-Link(RS485) and Ethernet to connect sensors (multi sensor system), actuators and several measurement devices with the computer system. In a typical configuration about 50-60 sensors and actuators are attached to the engine under test. Sensors connected via the IEEE1394 include: PT100, several high temperature sensors NiCrNi, DMS measurement, current, voltage, pressure and speed sensors.

More complex physical parameters are determined with measurement devices for fuel consumption, oil consumption, diesel, emission and fast response devices (Sensiflow air consumption measurement, BlowBy compression bypass amount).

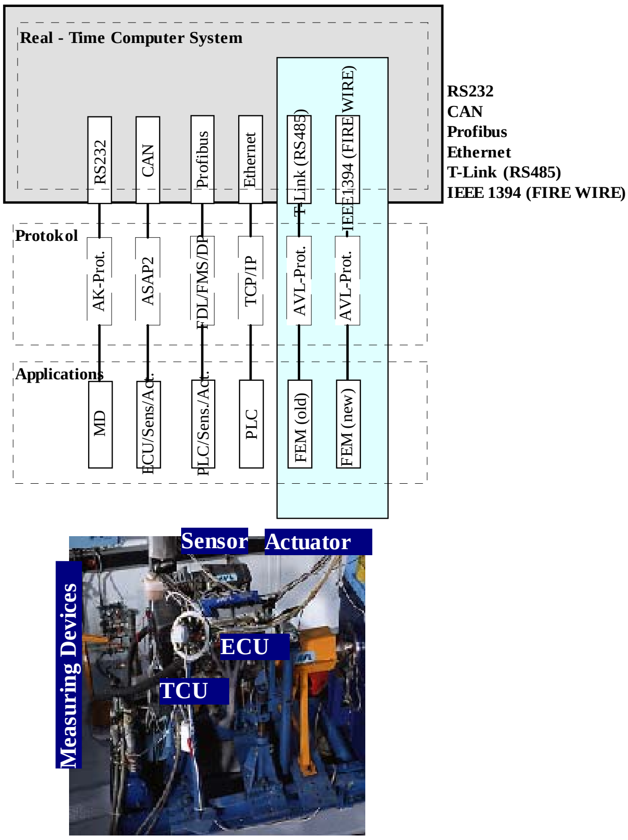

Figure 2: Part of the PUMA Open instrumentation interface

The PUMA Open is a complex object-oriented system and it combines both real-time (RT) and non real-time (NRT) computing on the same platform (PC solutions). The NRT part is based on the operating system Microsoft® Windows NT/2000/XP and the RT part is based on the Windows real-time extension INtime by TenaSys®. Its main components are:

- PUMA Operating System
- Control and Automation Functions
- Data Acquisition and Storage
- Multi-Level Safety Monitoring
- Graphical User Interface

The PUMA Open real-time computer system (RTCS) is based on a layered architecture. At the bottom lies the real-time operating system (RTOS) INtime. The next layer is ARTE (AVL Real-Time Environment), and on top of the architecture are the various PUMA sub-systems. ARTE provides all real-time services that are required by the other components. The ARTE services can be used via a standardized interface. All real-time tasks have priority over any non real-time processing in PUMA. Furthermore ARTE can be seen as a wrapper over the INtime RTOS and simplifies the development of real-time software components for the developers by providing customized real-time services as a library.

We distinguish between two types of configurations of the PUMA Open AS. The PUMA Open AS can be assembled using a (sub)set of all possible sub-systems and interfaces. Since this set of components does not change during operation of the AS, this is referred to as static configuration. During operation of the AS, several components may be activated and shut down, i.e., the AS is operated in different modes of operation. This is referred to dynamic configuration of the AS.

At a high-level view the PUMA Open AS has three different modes of operation. In the monitoring mode, only the PUMA operating system and the graphical user interface are activated. In this mode, the system is initialized, the system parameters are checked and loaded as well as the I/O sub-systems are booted. In the manual mode, the data acquisition and storage as well as the multi-level safety monitoring sub-systems are also activated. In this mode, the test and the engine (technical process) parameters are checked and loaded, the engine monitoring, the data acquisition, the limit monitoring and the post processing are activated. Finally, in the automatic mode all sub-systems are activated and an automatic test run is executed.

III. MODELING RT AND NRT PERFORMANCE

This paper focuses on the performance evaluation of the automation system PUMA Open. The modular and open platform design of PUMA Open enables the use of standardized interfaces, hardware and software. Therefore this AS consists of the operating system Microsoft® Windows (NT, 2000 or XP) for the NRT part and TenaSys® INTime for the RT part. The current state of the art concerning PUMA Open is a well working performance model for the real-time part: *PO-PEP*$_E$ *- PUMA Open - Performance Evaluation and Prediction Environment*.

PO-PEPE has been developed by D. Prisching [3]. The timing behavior of RT tasks can be analyzed with this tool based on a response time analysis (RTA). The RTA is a simple and effective procedure to model most aspects of a real-time system with defined priorities [2, 4, 5, 6]. Core of this computation is the calculation of the timing metrics cycle time (T_i), deadline (D_i) and computation time (C_i) of a set of tasks.

The RTA was extended for PUMA Open by RT-NRT interoperability. The real-time part of PUMA Open is well modeled with this analysis Tool. By the exact analysis of each RT-thread we also receive the data of the NRT-thread that represents the NRT operating system and its running applications (Windows thread). When no RT thread is ready to execute, the NRT thread is scheduled. The achieved values for the NRT thread are currently compared to empirical values for feasability tests.

Related Work

Until now there is no other research work known that models and predicts in detail the timing behavior of a combined Windows-INTime system. The main problem of this combined model is that due to the complexity of the OS Windows and the underlying hardware its timing behavior is almost non-deterministic. Thus, developing an accurate performance model would be infeasible. As an alternative often a statistical approach is applied.

There is some work available which deals with improving the determination of the worst-case execution time (WCET) on high-performance architectures:

Colin et al. [8] studied the influence of pipelines, caches, branch prediction units and out-of-order execution units on the worst-case execution time (WCET). They investigated that these speculative hardware features impact the WCET positively. By dividing the whole application in small chunks, called basic blocks, the WCET can be calculated more accurately. Furthermore, they state that WCET can also be regarded as a parametric function depending on the input and not as previously done as a static parameter.

Audsley and Bletsas [9] have investigated the effect of co-processors on the WCET. Typically, the CPU waits until the co-processor has finished its computation. However, there are situations where CPU and co-processor work concurrently. In these cases, the WCET can be reduced.

Traditional approaches for WCET produce pessimistic values for modern processors. For most of the analytical as well as measured data it is unlikely that certain levels of WCET are exceeded. Bernat [10] proposes that the execution time for instructions should no longer be constant. Deviations should be classified into data-dependent (multiplication, division), history-dependent (cache, pipelines, branch prediction) and mixed type (out-of-order execution) instructions.

Edgar and Burns [11] have also developed a statistical model for computing the WCET.

IV. PERFORMANCE ESTIMATION TOOL

The core feature of this tool is the estimation of the usability of a specified PUMA Open configuration. The user will be prompted to specify the planned system with all parameters. These parameters reflect the complete system under test beginning with the basic configuration (which parts are available and licensed in this system). The next level of parameters represents the current test configuration (type of testbed, number of I/O channels, …). The highest level of parameters specifies a so called performance scenario. The performance scenarios describe the actual use of possible functions and modules (number of graphic windows, recording function, filters, …). Given this system configuration a real-time system (RTS) analysis is performed. The RTS analysis uses a worst-case execution time calculation described in [3]. The results show whether all tasks can be executed in time. Furthermore, scheduling and the time slices given to the Windows-Task will be calculated. The Windows-Task properties like computation time and suspension time together with other factors like overall CPU load will have to meet some criteria which will be defined in the next steps of this work and will also be influenced by the selected system configuration.

Database

To support the work of the decision maker the usability estimation program will be embedded within a Microsoft Access database using Visual Basic for Applications. MS Access is widespread available and not bound to a network connection to a database server and it may even be copied to a laptop taken to a customer. A database is essential for this tool as many different configurations and settings will be used. The stored data and configurations will provide the source for easing the configuration selection process. Furthermore, most calculations will use the stored system analysis data so that only for new releases of PUMA Open this data has to be updated. Most parts of the configurations will not change rapidly but only the combination of different configurations. Therefore, many datasets will only change rather slowly.

Configuration and updating the program

The program will provide an import and update functionality. By these functions it is easily practicable to keep the database up-to-date with the latest changes in analysis and equipment configurations. Also transferring complete sets of system evaluations to another user’s database may be possible.

The following data packages will be imported separately to configure the program or to prepare a specific evaluation configuration:

- performance scenarios
- PC performance data
- PUMA Open configuration and parameter settings
- reference performance data

To model the system behavior major functional scenarios (focus scenarios), which are important from a performance perspective, are identified. These scenarios are defined as performance scenarios, which describe different representative possible cases of combinations of used modules and functions. Such performance scenarios could be the initial state with all I/O systems powered and running or a complete automatic test run classifying and recording 60 channels with intensive graphic display.

The PC performance data is a combination of a set of measurements performed on a dedicated PC hardware and a relating factor to the actual hardware, i.e., the comparison is based on workload benchmarks.

The PUMA Open configuration and parameter settings consist of several hundred values. These values mainly describe the number and type of different I/O bus systems such as having a CAN bus with 8 systems running at 500Hz.

The reference performance data is the result of a RT and NRT performance analysis. One of the primary goals of this work is to determine and model the performance data. As described in [3] each PUMA Open subsystems can be decomposed into defined modules. Each of these modules has a certain execution time (C_i) for a piece of code as well as a cyclic time (T_i). These modules are termed as *Extreme Piece of Code* (EPOC) and since C_i/T_i is the fraction of processor time spent in executing task τ_i, the utilization factor for n EPOCs is given by:

$$U = \sum \frac{C_i}{T_i}$$

With RTA further timing characteristics as response time (R_i) (at the critical instant or at multiple invocations) can be calculated.

So, it is necessary that all relevant subsystems (with their tasks and processes) are listed here,. They are analyzed regarding their EPOC's, and establish the relationship between EPOC's of the subsystems vs. their configuration and parameter settings. This approach determines and models the RT part of the PUMA Open system [3]. Nevertheless a practical approach to determine and model NRT parts must be established.

Output

The final results are estimates to support a decision whether a PUMA Open configuration is feasible. There is no guarantee that the designed configuration will always work in the desired manner concerning the user interface. However, if the tool reports a negative decision the planned configuration is not realizable.

In general the output will consist of a list of parameters showing the results of the response-time analysis as well as the WCET behavior describing the RT system (computation time, cyclic time, suspension time, response time, load and Windows-Task parameters). A violation of the real-time requirements is decided as system malfunction. From the Windows-task parameters together with the chosen configuration the NRT system is estimated and compared to different constraints such as the longest suspension time for Windows must be less than 7ms. This constraint checks whether a smooth human-computer-interface interaction is possible. If the estimation does not encounter any conflicts the system will be rated by a score how close to a conflicting situation it is. Finally an overview of the memory use especially for the RT-NRT shared memory is given.

V. CONCLUSIONS

In this paper we have presented our approach to evaluate and predict the performance in a complex automation system. Performance evaluation and prediction is an important part for the configuration support not only of complex automation systems but also for general computing systems. Although our performance model has been targeted to the PUMA Open automation system, a general approach to model the performance of complex computing systems can be derived. Future work will include the experimental evaluation of our performance estimation tool.

VI. REFERENCES

[1] D. Prisching, M. Paulweber, and B. Rinner, “Configuring Complex Multi-Sensor Test Bed Systems”, *Proceedings of the Telecommunication and Mobile Computing – Workshop on Wearable Computing,* Graz Austria 2003

[2] G. Bernat, “Response Time Analysis of Asynchronous Real-Time Systems”. *Real-Time Systems* 25(2-3): 131-156 (2003)

[3] D. Prisching, *Performance Evaluation and Prediction in Complex Real-Time Automation Systems*, PhD Thesis, Graz University of Technology, 2003

[4] N. C. Audsley and A. Burns, “Applying New Scheduling Theory to Static Priority Preemptive Scheduling”. Software Engineering Journal, 1993

[5] A. Burns, “Preemptive Priority-Based Scheduling: An Appropriate Engineering Approach”. Advances in RealTime Systems: 225-248, 1994

[6] K. Tindell, A. Burns and A. Wellings, “An Extendible Approach for Analyzing Fixed Priority Hard Real-Time Tasks”. 1992

[7] G. Bernat, A. Colin and S.M. Petters, “WCET analysis of probabilistic hard real-time systems”. In Proceedings of the 23rd Real-Time Systems Symposium RTSS 2002, pages 279-288, 2002

[8] G. Bernat, A. Colin and S.M. Petters, “pWCET: a tool for probabilistic worst case execution time analysis of real-time systems”. Technical report YCS353, University of York, 2003

[9] N. C. Audsley and K. Bletsas, “Fixed Priority Timing Analysis of Real-Time Systems with Limited Parallelism”. Proceedings of the 16th Euromicro Conference on Real-Time Systems (ECRTS 04), 2004

[10] G. Bernat, A. Colin and S. M. Petters, “WCET Analysis of Probabilistic Hard Real-Time Systems”. 23rd IEEE Real-Time Systems Symposium (RTSS'02), 2002

[11] R. Edgar and A. Burns, “Statistical Analysis of WCET for Scheduling”, 22nd IEEE Real-Time Systems Symposium (RTSS’01), 2001