A Statistical Approach to Simulation Model Validation in Timing Analysis of Complex Real-Time Embedded Systems

Yue Lu, Johan Kraft, Thomas Nolte and Christer Norström
Mälardalen Real-Time Research Centre
Mälardalen University, Västerås, Sweden
{yue.lu, johan.kraft, thomas.nolte, christer.norstrom}@mdh.se

Abstract

Simulation-based analysis methods make few restrictions on the system design and scale to very large and complex systems, therefore they are widely used in timing analysis of complex industrial embedded systems. This paper presents a statistical approach to validation of temporal simulation models extracted from complex real-time embedded systems, by introducing existing mature statistical methods to the context. The proposed approach first collects sampling distributions of response time and execution time data of tasks in both the modeled system and the model, based on simple random samples (SRS). The second step of the approach is to compare the sampling distributions, regarding interesting timing properties, by using the non-parametric two-sample Kolmogorov-Smirnov test. The evaluation using a fictive system model inspired by a real robotic control system with a set of change scenarios, shows a promising result. The proposed algorithm can identify temporal differences between the target system and its extracted model, i.e., the algorithm can assess whether the extracted model is a sufficiently accurate approximation of the target system.

1 Introduction

To date, most existing embedded real-time software systems have been developed in a traditional code-oriented manner, over extended periods of time, sometimes spanning decades. As a result, many such systems become large and increasingly complex. Further, to maintain, verify and reuse these systems is difficult and expensive. There are many industrial embedded systems having a very complex runtime behavior, due to that they are highly configurable and event-triggered. Such systems consist of millions of lines of C code, and contain 50 - 100 tasks or more, out of which many tasks have real-time constraints. One example of such systems is the robotic control systems developed by ABB [1]. Further, the temporal dependencies between tasks in such systems vary the execution time and response time of tasks radically. We refer to such systems as *Complex Real-Time Embedded Systems* (CRTES).

Simulation-based analysis of CRTES has the potential of not only allowing for response-time analysis of such systems [2], [3], but also facilitating migration toward a component based real-time system by e.g., analyzing the timing properties of the existing code and wrapping it into components. Moreover, simulation-based methods can also be used in timing impact analysis [4], i.e. to analyze the impact of changes on a system's temporal behavior, before introducing changes to the system.

A major issue when using simulation-based timing analysis is how to obtain the necessary analysis model, which should be a subset of the original software program focusing on behavior of significance for task scheduling, communication and allocation of logical resources. For many systems, manual modeling would be far too time-consuming and error-prone. Two methods for automated model extraction are proposed in [5]. A tool for automated model extraction is in development, named MXTC - Model eXtraction Tool for C. The MXTC tool targets large implementations in C, consisting of millions of lines of code, and is based on program slicing [6]. The output of MXTC is simulation models for the RTSSim simulation framework [7].

However, there is one important issue to be raised, i.e. *model validity*, which is defined as *the process of determining whether a simulation model is an accurate representation of the system, for the particular objectives of the study* [8]. As a model is an abstraction of the system, some system details may be omitted in the model, for instance when using probabilistic execution time modeling. Thus, the results from a simulation of such models may not be identical to the recordings of the system, e.g., with regard to the exact task response time. In order to convince system experts to use simulation-based methods, the models should reflect the system with a satisfactory level of significance, i.e., as a sufficiently accurate approximation of the actual system. Moreover, other threats to model validity are the configuration of the model extraction tool and bugs in the model extraction and analysis tools. Therefore, an appropriate validation process should be performed before using the models.

In this paper, we present a statistical approach for validation of temporal simulation models extracted from real industrial control systems containing intricate task execution dependencies. That is, to consider this particular problem as a statistical problem, then, which could be solved by using existing, mature methods from the field of statistics.

The proposed method *StatiVal* collects sampling distributions by combining using simple random samples (SRS) [9] with our presented mechanism to eliminate dependencies among raw Response Time (RT) and Execution Time (ET) data caused by task execution dependencies in the system. Next, our method will produce results concerning whether the model is a sufficiently accurate approximation of the target system, from the perspective of relevant timing properties such as response time and execution time of tasks in the modeled system and the extracted model, by using the non-parametric two-sample Kolmogorov-Smirnov test [10]. Since our tool for model extraction (MXTC) is not yet ready, in this work, we evaluate StatiVal by using a manually created simulation model inspired by an industrial robotic control system. Then, the original model is compared with different variants of the model, each of which variant corresponds to a particular change scenario. Our evaluation of this method shows the promising results, i.e., StatiVal can identify timing differences between the modeled system and models, and should be applicable in a non-trivial industrial evaluation and deployment of our framework for simulation-based analysis.

The remaining part of the paper is organized as follows: Section 2 introduces the simulation model used in this work. Section 3 presents the related work about model validation at first, and then gives problem formulation, descriptive statistics of raw RT and ET data of tasks in the evaluation model, and the problems with using parametric statistics, respectively. Section 4 and Section 5 introduce our proposed method and evaluation results, and finally, Section 6 concludes the paper and discusses future work.

2 RTSSim Simulation Models

The proposed validation method primarily targets simulation models for the RTSSim simulation framework, which is quite similar to *ARTISST* [11] and *VirtualTime* [12]. An RTSSim simulation model consists of a set of tasks, sharing a single processor. Each task in RTSSim is a C program, which executes in a "sandbox" environment with similar services and runtime mechanisms as a normal real-time operating system, e.g., task scheduling, inter-process communication (message queues) and synchronization (semaphores). The default scheduling policy of RTSSim is Fixed-Priority Preemptive Scheduling (FPPS) and each task has scheduling attributes such as priority, period, offset and jitter. RTSSim allows for three types of selections which are directly controlled by simulator input data: Selection of execution times in `execute` statements; Selection of task jitter; Selection of task behaviors, depending on the system environment, e.g., random number of external events generated by sensors. In RTSSim, Monte Carlo simulation is realized by providing randomly generated input data. A more thorough description of RTSSim can be found in [7].

3 Model Validation

3.1 Related Work

For the sake of space, we only briefly introduce the related work concerning the model validation process. There are various methods to do the comparison; these methods are either objective or subjective. Subjective methods are often used for validation of simulation models; examples of subjective methods are Face Validation, Graphical Comparisons and Sensitive Analysis [13], which are highly dependent on domain expertise and hence error-prone. Objective methods use mathematical methods to compare outputs from the real system with output from the simulation model. In [14], the authors presented a notation of model equivalence based on observable property equivalence which is used to compare results of a model and an actual system. A method in [15] is presented for automated validation of models extracted from real-time systems by checking if the model can generate the same event sequences as the recorded event sequences from the system using a model checker.

3.2 Problem Formulation

We are given a model $S^{'}$ which is extracted from a real system (or modeled system) S containing a task set Γ including n tasks, where $n \in \mathbb{N}$. Let $RT_{samples}(S^{'}, \tau_i)$, $RT_{samples}(S, \tau_i)$, $ET_{samples}(S^{'}, \tau_i)$ and $ET_{samples}(S, \tau_i)$ denote the sampling distributions of the response time and execution time measured for a task τ_i in $S^{'}$ and S respectively. The goal of the problem is then to find: whether there are statistically significant differences between the system and model distributions with respect to response times and execution times of the adhering tasks, or can they be considered statistically equal (i.e., from the same population).

3.3 Descriptive Statistics of Raw RT and ET Data

Table 1 shows the numerical summary of the center and the spread (or variability) of sampling distributions of the response time (RT) data of tasks in Model 1 (M1) containing intricate execution dependencies, used for the evaluation in Section 5. In Table 1, *Std. Dev*, *Q1* and *Q3* represents *standard deviation*, *first quartile* and *third quartile* of the sampling distribution respectively. As we can see, the skewness of sampling distributions for all the tasks except for the IO task are right (positive) skewed (i.e., the numerical representation of tasks' skewness are positive; in the view of graph, the sampling distribution has relatively few high values, and the mass of the distributions is concentrated on the left of the figure). Further, the outliers existing in raw RT data as well as ET data of all tasks cannot be removed since they are not generated due to system errors or hardware failures. Therefore, we have the reasoning to add the *five-number summary* introduced in [9] consisting of *Min*, *Q1*, *Median*, *Q3* and *Max* to Table 1. Due to limited space, we only show the sampling distribution of raw RT data of one task i.e., the CTRL task when the number of samples is large enough i.e. 199 990 in one simulation run (refer to row `Samples` for the CTRL task in Table 1), as an example shown in Figure 1. Further, note that the outliers in the picture might not be clear enough to see, though in fact, they approximately exist in the range of [3 000, 6 829] along with the horizontal axis.

Table 1. Descriptive statistics of sampling distributions of raw RT data of tasks in the system model M1 used in the evaluation.

	DRIVE	*IO*	*CTRL*	*PLAN*
Samples	199994	400000	199990	199988
Mean	222.08	125.0	1967.3	2002.9
Std. Dev	14.291	45.576	389.98	412.46
Skewness	6.7334	0.00128	0.38184	7.0644
Min	220	0	1024	332
Q1	220	100	1594	1631
Median	220	125	1919	1931
Q3	220	150	2339	2376
Max	420	250	6829	45957

3.4 Dependencies between Raw RT and ET Data of Tasks

In our case, due to intricate task execution dependencies in the system, an upcoming RT data may not be independent with the RT data previously recorded at each simulation run (we refer to such RT and ET data as *raw RT and ET data*). The same problem applies for raw ET data. Second, in the conventional statistical procedure (*parametric test*), e.g., t-test, analysis of variance (ANOVA) [16], one important assumption is that the underline population is assumed to follow a normal distribution. However, such assumption cannot be made since the sampling distribution of either raw RT data or raw ET data of all tasks often is conforming to a multimodal distribution having several peaks (consider Figure 1 as an example). Specifically, because of such distinctive feature of our target industrial control system, it is difficult to bring conventional statistical methods into the context. A new way of constructing the sampling distributions of tasks' RT and ET data has to be introduced, in order to fulfill the basic requirement given by *probability distribution*, i.e. the variable described by a probability distribution is a `random variable`, of which value is a function of the outcome of a statistical experiment that has outcomes of equal probability. We will present the proposed mechanism in the following Section 4.2.

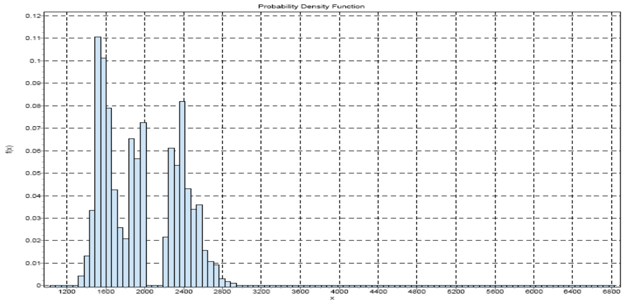

Figure 1. The sampling distribution of raw RT data of the CTRL task in the evaluation model M1.

4 Algorithm

4.1 Simple Random Samples

In order to eliminate bias on the sampling, which is a key issue of selecting samples from the population of all individuals concerning the desired information, the technique of simple random samples (SRS) [9] is adopted. SRS gives every possible sample of a given size the same chance to be chosen. For instance, Monte Carlo simulation is used as a way of implementing SRS to collect sampling distributions of RT and ET data of tasks in the extracted RTSSim model. This is done by an embedded random number generator `rnd_inst()` in the RTSSim simulator, which is an improved version of the Pseudo-random number generator used in C, i.e., `rand()` in Algorithm 1. The detailed implementation of `rnd_inst()` is shown in Algorithm 1. Moreover, empirical results showed that the distribution of ran-

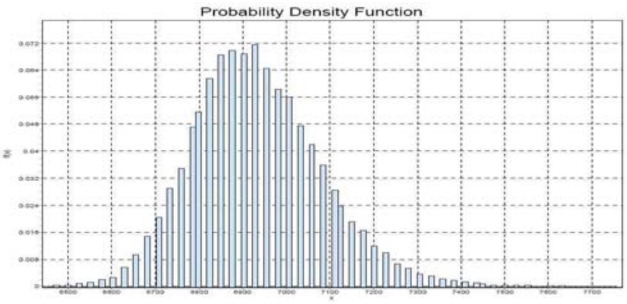

Figure 2. A new reconstructed sampling distribution of RT data of the CTRL task in the evaluation model M1.

dom numbers given by `rnd_inst()` is conforming to the uniform distribution, which assures that for each selection in RTSSim input data, all possible values in any range are equally likely to be chosen. Analogously, the sampling distributions of RT and ET data of tasks in the real system can be collected based on measurements given a randomized system input. Some of the outliers (extreme values) which are caused, e.g. hardware failure or system errors, have to be removed from the sampling distributions.

Algorithm 1 *rnd_inst*()

$temp1 \leftarrow rand()$
$temp2 \leftarrow rand()$
$ret \leftarrow temp1 \times 32768 + temp2$
return ret

4.2 Reconstruction of New RT and ET Sampling Distribution

In order to eliminate dependencies between raw RT and ET data of tasks due to intricate task temporal dependencies, we propose a method by first running N Monte Carlo simulations conforming to SRS as introduced previously. Further, for each task in the task set Γ, the highest value of m samples RT data and m samples ET data recorded by each simulation, will be chosen to construct new sampling distributions of RT data and ET data. By doing this, the new constructed sampling distributions of RT and ET data of tasks can be considered from a random variable, since there are no dependencies between any maximum value of RT and ET data of tasks between two independent simulations. In other words, task intricate temporal dependencies are kept in new sampling distributions of RT and ET data, while the dependencies between any RT data and ET data are eliminated. Refer to Figure 2 as an example.

4.3 Problems with Using Parametric Statistics

So as to determine if the conventional statistical procedure (*parametric test*), e.g., t-test, ANOVA, can be applied to infer parameters of new tasks' RT and ET sampling distributions used for validation purpose, the conclusion, that if such sampling distributions[1] are from a normal distribution, has to be drawn at first. In this work, it is done by using a commercial statistic analysis software *EasyFit* [17], according to the results given by a Goodness of Fit (GOF) test, i.e., Chi-squared test at α-value of 0.05[2]. The obtained results clarify that new sampling distributions of RT and ET data of all tasks do not conform to any of the 65 known distributions, e.g., Normal, Uniform, Student's t, Lognormal. The null and alternative hypotheses used in Chi-squared test, at significance level 0.05, are as follows.

1. H_0: the sampling distribution concerning the RT or ET data of task τ_i follows a specific distribution;
2. H_a: the sampling distribution concerning the RT or ET data of task τ_i does not follow a specific distribution.

Note that the 65 known distributions can be found in [17]. Further, in t-test, the mean value μ_0 of the population has to be known beforehand, which is not the fact in our case. Because a parametric test cannot be reasonably applied in this work, we thereby use the two sample Kolmogorov-Smirnov (hereafter KS test) which is nonparametric and makes no assumptions on the underline population of a sampling distribution.

4.4 StatiVal

The proposed method, *StatiVal*, is shown in Algorithm 2. The algorithm returns the result concerning if there exist a statistically significant difference between the two data sets that are from the modeled system S and the model S', in the view of system timing properties including tasks' response time and execution time. Further, in this work, since we cannot perform the validation between the real modeled system and the extracted model, we will instead compare a system model S inspired by a real industrial robotic control system (considered as the modeled system) with a set of models S' where a specific change scenario (as shown in Table 3) is applied. Both of S and S', are in this case simulation models, analyzed using Monte Carlo simulation which in Algorithm 2 is modeled as a function, *MTC*, with four parameters: m - the number of samples drawn from each simulation trace, τ_k - the task on focus in KS test, *Property* - either RT or ET of the task τ_k and *rnd_inst*() - a random number generator in RTSSim simulator. When the reference for comparison is a real system, the sampling distribution is built by using random measurement (e.g., by randomizing inputs to the system) at first, and then removing

[1] In our case, the number of samples i.e., 20 000 in sampling distributions of RT and ET data of tasks is statistically enough to represent the underline population.

[2] α = 0.05 means that we are requiring that the RT and ET data of tasks give evidence against H_0 so strong that it would happen no more than 5% of the time when H_0 is true.

outliers from the sampling data that are caused by hardware failure or system errors during each system runtime observation, and finally, choosing the highest value of RT and ET data of tasks in the system. Further, because such activity is also application-specific, we therefore will not discuss it in details in this work. The outline of StatiVal is as follows:

1. Construct the sampling distribution of N RT and ET data of all the tasks in both the system S and the model S' by Monte Carlo simulation `MTC()` respectively (refer to lines 1 to 16 in Algorithm 2).
2. Use KS test to compare if sampling distributions of RT and ET data of each task τ_k in the task set Γ in both S and S' are statistically significant iteratively. If the result given by KS test is H_a, then Algorithm 2 draws the conclusion C_1, i.e. *the model S' is not a sufficiently accurate approximation of the system S due to an improper model extraction process*, and finally, stops the validation process; Otherwise, the entire validation process will terminate after all the tasks are evaluated by KS test (refer to lines 18 to 33 in Algorithm 2). In practice, KS test is conducted by using a commercial software *XLSTAT* [18], which is a plug-in to EXCEL and returns the result by comparing two sampling distributions containing 20 000 samples per each, in a few seconds.

Algorithm 2 $StatiVal(\Gamma)$

1: **for all** τ_k such that $1 \le k \le n$ in Γ in both S and S' **do**
2:   **for all** i such that $1 \le i \le N$ **do**
3:     $X_i \leftarrow x_{i,1}, ..., x_{i,j}, ..., x_{i,m} \leftarrow MTC(m, \tau_k, RT, rnd\_inst())$
4:     $X_{\tau_k,i} \leftarrow Max(X_i)$
5:     $Y_i \leftarrow y_{i,1}, ..., y_{i,j}, ..., y_{i,m} \leftarrow MTC(m, \tau_k, ET, rnd\_inst())$
6:     $Y_{\tau_k,i} \leftarrow Max(Y_i)$
7:     $X'_i \leftarrow x'_{i,1}, ..., x'_{i,j}, ..., x'_{i,m} \leftarrow MTC(m, \tau_k, RT, rnd\_inst())$
8:     $X'_{\tau_k,i} \leftarrow Max(X'_i)$
9:     $Y'_i \leftarrow y'_{i,1}, ..., y'_{i,j}, ..., y'_{i,m} \leftarrow MTC(m, \tau_k, ET, rnd\_inst())$
10:     $Y'_{\tau_k,i} \leftarrow Max(Y'_i)$
11:   **end for**
12:   $X_{\tau_k} \leftarrow X_{\tau_k,1}, ..., X_{\tau_k,i}, ..., X_{\tau_k,N}$
13:   $Y_{\tau_k} \leftarrow Y_{\tau_k,1}, ..., Y_{\tau_k,i}, ..., Y_{\tau_k,N}$
14:   $X'_{\tau_k} \leftarrow X'_{\tau_k,1}, ..., X'_{\tau_k,i}, ..., X'_{\tau_k,N}$
15:   $Y'_{\tau_k} \leftarrow Y'_{\tau_k,1}, ..., Y'_{\tau_k,i}, ..., Y'_{\tau_k,N}$
16: **end for**
17: $ret \leftarrow 0$
18: **for all** τ_k such that $1 \le k \le n$ in Γ in both S and S' **do**
19:   $ret \leftarrow kstest(X_{\tau_k}, X_{\tau'_k}, \alpha)$
20:   **if** $ret = H_0$ **then**
21:     $ret \leftarrow C_0$
22:   **else**
23:     $ret \leftarrow C_1$
24:     **return** ret
25:   **end if**
26:   $ret \leftarrow kstest(Y_{\tau_k}, Y_{\tau'_k}, \alpha)$
27:   **if** $ret = H_0$ **then**
28:     $ret \leftarrow C_0$
29:   **else**
30:     $ret \leftarrow C_1$
31:     **return** ret
32:   **end if**
33: **end for**
34: **return** ret

5 Evaluation

5.1 The Evaluation Model

Currently, we are not able to perform the model validation process concerning the extracted model and a real system. Therefore, in this work, we examine the idea by using a simulation model Model 1 (M1) describing a fictive, representative industrial robotic control system developed by ABB. It is designed to include some behavioral mechanisms from the ABB system:

1. tasks with intricate dependencies in temporal behavior due to Inter-Process Communication (IPC) and globally shared state variables;
2. the use of buffered message queues for IPC, which vary the execution time of tasks dramatically;
3. although FPPS is used as base, one task, i.e., the CTRL task, changes its priority during runtime, in response to system events.

Further, the task model is presented in Table 2. The details of the model are described in [7].

5.2 Change Scenarios and Results

The RT and ET data of tasks produced by the original simulation model M1 is used as reference, for comparing the impact of a set of *change scenarios* which are initially introduced in [19] and outlined in Column *Changes Description* in Table 3. Moreover, for Case 4, 5 and 6, there is a DUMMY task added to the model S' with different priorities, execution times and periods (denoted as C and T in Table 3 respectively). Finally, we compare the outputs against the original model to investigate the performance of the method. The results given by StatiVal are shown in Table 3, which are in line with the expected results in [19]. More importantly, our evaluation shows a promising result, i.e. the proposed algorithm can identify temporal differences between the target system and its extracted model by showing the evidence whether the extracted model is a sufficiently accurate approximation of the target system.

6 Conclusions and Future Work

This paper has presented our work on validation of temporal simulation models extracted from real industrial control systems containing intricate task execution dependencies. In particular, we have presented and evaluated the method by using a fictive system model inspired by a real

Table 2. Tasks and task parameters for M1. The lower numbered priority is more significant, i.e., 0 stands for the highest priority.

Task	*Period (μs)*	*Offset (μs)*	*Priority*
DRIVE	2000	12000	2
IO	5000	500	5
CTRL	10000 or 20000	0	6 or 4
PLAN	40000	0	8

Table 3. Results obtained by using StatiVal concerning different models according to change scenarios.

Change Scenarios	*Changes Description*	*RT*	*ET*	*StatiVal*
Case 1	IO: C 23 → 46	H_a	H_a	C_1
Case 2-1	PLAN: Prio 8 → 9	H_0	H_0	C_0
Case 2-2	PLAN: Prio 8 → 3	H_a	H_a	C_1
Case 3-1	PLAN: T 40 000 → 80 000	H_0	H_0	C_0
Case 3-2	DRIVE: T 2 000 → 10 000	H_a	H_a	C_1
Case 4-1	DUMMY: Prio = 7, T = 5 000, C = 25	H_a	H_0	C_1
Case 4-2	DUMMY: Prio = 7, T = 5 000, C = 50	H_a	H_0	C_1
Case 5-1	DUMMY: Prio = 1, T = 5 000, C = 25	H_a	H_a	C_1
Case 5-2	DUMMY: Prio = 1, T = 5 000, C = 50	H_a	H_a	C_1
Case 6-1-1	DUMMY: Prio = 1, T = 10 000, C = 50	H_a	H_a	C_1
Case 6-1-2	DUMMY: Prio = 1, T = 10 000, C = 100	H_a	H_a	C_1
Case 6-2-1	DUMMY: Prio = 7, T = 10 000, C = 50	H_a	H_0	C_1
Case 6-2-2	DUMMY: Prio = 7, T = 10 000, C = 100	H_a	H_0	C_1

system with a set of change scenarios, which shows that the proposed method has the potential to identify temporal differences between the modeled system and the extracted models. As part of future work, an effort will be spent on evaluating more scenario changes on the evaluation model. Moreover, we will evaluate the method on real systems.

References

[1] "Website of ABB Group," www.abb.com.

[2] J. Kraft, Y. Lu, C. Norström, and A. Wall, "A metaheuristic approach for best effort timing analysis targeting complex legacy real-time systems," in *RTAS 08*, April 2008, pp. 258–269.

[3] M. Bohlin, Y. Lu, J. Kraft, P. Kreuger, and T. Nolte, "Simulation-based timing analysis of complex real-time systems," in *RTCSA 09*, August 2009, pp. 321–328.

[4] J. Andersson, J. Huselius, C. Norström, and A. Wall, "Extracting simulation models from complex embedded real-time systems," in *ICSEA'06*. IEEE, 2006.

[5] J. Kraft, J. Huselius, A. Wall, and C. Norström, "Extracting simulation models from complex embedded real-time systems," in *Real-Time in Sweden 2007*, August 2007.

[6] M. Weiser, "Program Slicing," in *ICSE '81*. IEEE Press, 1981, pp. 439–449.

[7] J. Kraft, "RTSSim - A Simulation Framework for Complex Embedded Systems," Mälardalen University, Technical Report, March 2009.

[8] A. M. Law, "How to build valid and credible simulation models," in *WSC '08*. Winter Simulation Conference, 2008, pp. 39–47.

[9] D. S. Moore, G. P. Mccabe, and B. A. Craig, *Introduction to the practice of statistics*, 6th ed. New York, NY 10010: W. H. Freeman and Company, 2009.

[10] A. M. Law and D. M. Kelton, *Simulation Modeling and Analysis*. McGraw-Hill Higher Education, 1999.

[11] D. Decotigny and I. Puaut, "ARTISST: an extensible and modular simulation tool for real-time systems," in *ISORC '02*, 2002, pp. 365–372.

[12] "Rapita systems, www.rapitasystems.com, 2008."

[13] O. Balci, "How to assess the acceptability and credibility of simulation results," in *WSC '89*. New York, NY, USA: ACM, 1989, pp. 62–71.

[14] J. Andersson, A. Wall, and C. Norström, "Validating temporal behavior models of complex real-time systems," in *SERPS'04*, September 2004.

[15] J. Huselius, J. Andersson, H. Hansson, and S. Punnekkat, "Automatic generation and validation of models of legacy software," in *RTCSA '06*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 342–349.

[16] "t-test and ANOVA, http://mathworld.wolfram.com, 2010."

[17] "Easyfit, www.mathwave.com/products/easyfit.html, 2010."

[18] "Xlstat, www.xlstat.com, 2010."

[19] F. Nemati, J. Kraft, and C. Norström, "Validation of temporal simulation models of complex real-time systems," in *CORCS'08*, July 2008.