

# 2014 Fifth International Conference on Information Science and Applications

# ICISA 2014

Technically Co-Sponsored by IEEE Computer Society

Seoul, Korea  
6–9 May 2014

Technically Co-Sponsored by





## Track 12: SATA Workshop

|  |            |
|--|------------|
| <b>A Study on Preventative Measures for Ninja Hacking .....</b>  | <b>552</b> |
| <i>Byungho Park, R. Young Chul Kim and Youn B. Park</i>  |            |
| <b>An Efficient I/O Interface Control Block Design Methodology for<br/>Application-Specific MPSoC Platforms .....</b>    | <b>555</b> |
| <i>Myoung-Seo Kim and Jean-Luc Gaudiot</i>   |            |
| <b>Applying MapReduce Programming Model for Handling Scientific Problems .....</b>                                       | <b>559</b> |
| <i>Yun Hee Kang and Young B. Park</i>  |            |
| <b>Design and Development of Multimedia Cultural Contents for Virtual Reality -<br/>Gogurye Ancient Tomb Mural .....</b> | <b>561</b> |
| <i>YuKyong Lee, Yong-Hwan Lee, Sung-Ho Hong,<br/>Youngseop Kim and Chang-June Kim</i>                                    |            |
| <b>Design of System Framework for Searching Medical Information .....</b>  | <b>563</b> |
| <i>Sung-Ho Hong, Youngseop Kim, Yong-Hwan Lee and Je-Ho Park</i>   |            |
| <b>Dynamic Configuration of SSD File Management .....</b>  | <b>565</b> |
| <i>Hyuk-Kyu Lim and Je-Ho Park</i>   |            |
| <b>Effective Fat Quantification Using Improved Least-Square Fit at High-Field MRI .....</b>                              | <b>568</b> |
| <i>Sung-Jong Eun and Taeg-Keun Whangbo</i>   |            |
| <b>Generation of Metadata Using Content-Based Image Retrieval System .....</b>   | <b>570</b> |
| <i>Sun-A Lee, Min-Uk Kim and Kyoungro Yoon</i>   |            |
| <b>Goal-Oriented Requirements Analysis Using Goal Accumulation RDF Graph .....</b>                                       | <b>572</b> |
| <i>Jun H. Lee and Young B. Park</i>  |            |
| <b>Image Recognition System That Uses Visual Word .....</b>  | <b>575</b> |
| <i>Min-Uk Kim and Kyoungro Yoon</i>  |            |
| <b>Implementation of a Continuous Playing System Based on RESTful Web Services .....</b>                                 | <b>577</b> |
| <i>Chompoo Suppatoomsin, Apiradee Ampawasiri and Cheong-Ghil Kim</i>   |            |
| <b>Metadata Schema for Augmented Reality .....</b>   | <b>579</b> |
| <i>Takaaki Ishikawa and Je-Ho Park</i>   |            |
| <b>Model Transformation Rule for Generating Database Schema .....</b>  | <b>581</b> |
| <i>Chae Yun Seo, R. Young Chul Kim and Young B. Park</i>   |            |
| <b>Modeling and Simulation for Embedded Software System .....</b>  | <b>584</b> |
| <i>So Young Moon, Bo Kyung Park, R. Young Chul Kim<br/>and Young B. Park</i>   |            |



# Modeling and Simulation for Embedded Software System

So Young Moon

Dept. of Computer & Information Communication, Hongik  
University, Sejong Campus, 339-701  
Korea  
[msy@selab.hongik.ac.kr](mailto:msy@selab.hongik.ac.kr)

R. Young Chul Kim

Dept. of Computer & Information Communication, Hongik  
University, Sejong Campus, 339-701  
Korea  
[bob@selab.hongik.ac.kr](mailto:bob@selab.hongik.ac.kr)

BO Kyung Park

Dept. of Computer & Information Communication, Hongik  
University, Sejong Campus, 339-701  
Korea  
[bk@selab.hongik.ac.kr](mailto:bk@selab.hongik.ac.kr)

Young B. PARK

Dept. of Computer Science, Dankook University, 330-714  
Korea  
[ybpark@dankook.ac.kr](mailto:ybpark@dankook.ac.kr)

**Abstract**— For productivity and correctness of embedded software system, I will focus on the verification of modeling by using M&S (Modeling & Simulation) in the modeling part. M&S (Modeling & Simulation) will be used to verify the modeling. If models are simulated for verification after modeling when we develop the embedded software, we may have the correctness of models. Also, it is probably going to improve embedded software's reusability and reliability because we will be reusing correctness models.

In this paper, we will extend a dynamic diagram of UML2.0 and xUML including Real-Time concepts to adapt to the embedded environment for embedded software modeling. Furthermore, it is probably going to improve reliability of embedded software system with using modeling and simulation concepts.

**Keywords**— *Embedded Software System; State Diagram; Concurrency; Stochastic; Modeling and Simulation*

## I. INTRODUCTION

Embedded software is a type of software that is built into hardware systems. This software is typically designed to perform and control one specific function. It is built in systems like cellular phones, televisions, planes, elevators, cars, robots, etc. Embedded software operates a lot of sensors in a car, and which controls flight control systems, navigation system in an UAV(Unmanned aerial vehicle). Embedded system such as nuclear power generations, aircraft controls, missiles needs high quality reliability because incorrect operation or suspend working is cause critical problems. Reactive system like embedded system has nested if-else statements, switch-case statements. David Harel's state chart is the best method for modeling a reactive system [1]. A state diagram models if-else, switch-case statements of complex embedded system and adopts various state patterns, so that is able to solve some problems before coding. In this paper, considering embedded system features, we extends dynamic diagram of UML 2.x included real-time concepts to embedded environment for modeling embedded software. Also we suggest stochastic

based state diagram according to probability state in embedded system that discovers new optimized transitions and states through stochastic state changes and state transition tables. Through this method, we can get correctness of modeling.

## II. RELATED WORK

### A. State Diagram

A state diagram is focus on state transition of single object occurred in consequence of an external event. Reactive system is an interactive system, which exchanges events with surrounding environments continuously. State Machine is the best method for modeling graphic user interface and embedded system [1]. State diagram describes possible states about a class and an event occur a state transition. State diagram is useful to show life cycle of a class.

### B. Concurrency

Concurrency handles multi-process at the same time in a single process. Concurrency programs process many processes that is in contrast with a sequential processing program by fixed order. For example, in a bridge only one car can pass through, if two car try to pass, nobody will pass or they will crush. This kind of problem also occurs in embedded system.

In this paper, we describe model with using LTS (Labeled Transition Systems)[2] state machine. Furthermore, LTS is described by FSP which is a text language.

#### 1) Deterministic

$(x \rightarrow P \mid y \rightarrow Q)$  presents action  $x$  or action  $y$  is ran collectedly. When action  $x$  is performed, process  $P$  will perform. When action  $y$  is performed, process  $Q$  will perform. " $\mid$ " is a select operator. Deterministic gives authority to a user what action will be performed.



TABLE I. FSP FOR COFFEE COIN MACHINE.

DRINKS = (red → coffee → DRINKS  
| blue → tea → DRINKS ).

### 1) Non-Deterministic

Non-deterministic gives authority to a system and action will be out of order execution. In case of deterministic has only one event from one state to other state, but in case of non-deterministic has more than one transition about one event.

TABLE II. FSP FOR COINTOSS

COIN = (toss → heads → COIN  
| toss → tails → COIN).

## III. EMBEDDED SOFTWARE SYSTEM MODELING

Dynamic modeling is important to model and simulate embedded software system. This chapter, we will present concurrent state diagram, stochastic based state diagram for complementing previous problem.

### A. Concurrent State Diagram

In case of HexAboider with two antenna sensors walks, if one antenna generates an event to move next transition, this robot is not able to walk. We suggest to use concurrency concepts for solving this kind of problem.

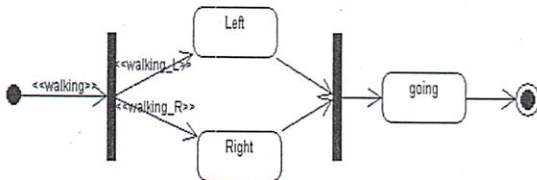


Fig. 1. Concurrent State Diagram for HexAboider.

### B. Stochastic Based State Diagram

General stochastic process means a collection of random variables; this is often used to represent the evolution of some random value, or system, over time. This is the probabilistic counterpart to a deterministic process (or deterministic system). In a stochastic or random process there is some indeterminacy: even if the initial condition (or starting point) is known, there are several (often infinitely many) directions in which the process may evolve. In the simple case of discrete time, as opposed to continuous time, a stochastic process involves a sequence of random variables and the time series associated with these random variables[3]. We apply the system design

with this concept. In the real world, it should predict to happen several possible behaviors than only single possible behavior, which may make unstable system. To solve this problem we suggest stochastic based state diagram to design the system which can't predict whether it can work to progress one way or not. That is, not unclear whether it moves from one state to other state or any state.

TABLE III. TYPE AND DESCRIPTION FOR STATE DIAGRAM

| Description  | Example of State Diagram |
|--|--------------------------|
| Stochastic: if at least one output with random value, next transition will be like a right figure. |                          |
| Concurrency: when occur several threads, and execute concurrently                                  |                          |

We make one example to explain the wiper rain sensors of the mobile car based on a stochastic system. The main functions of rain sensors sense the amount of rain to move a proper speed of wipers, and also stop the wiper without raining. If the rain falls down the surface of the car wind screen, Figure 2 represents the state diagram to work the behaviors of the car wiper based on unpredictable situation.

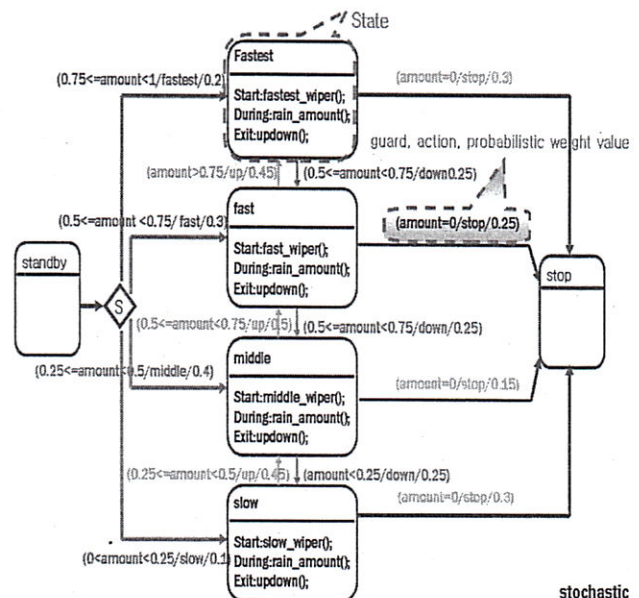


Fig. 2. Stochastic based State Diagram for Car Wiper.

The wiper doesn't work with predicting the amount of rain, but it work with measuring the amount of rain per time. So it is similar to non-deterministic, but it depends on stochastic state. Figure 3 shows that we test wiper state with LTSA.

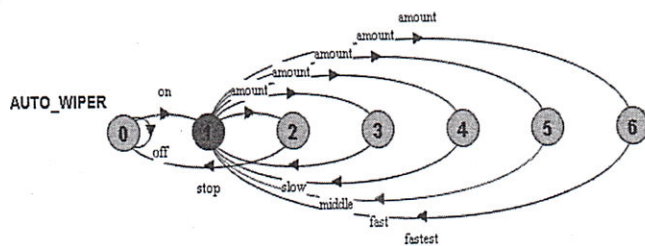


Fig. 3. LTS State Diagram for Car Wiper.

#### IV. CONCLUSION

This paper analyzes dynamic aspects of our suggested state diagram in UML x.x, to solve concurrent problems of a system. But in this time, the existing state diagrams lack of handling the events within the embedded software system and modeling time oriented area which also cannot handle stochastic aspects. To deal with these problems, we suggested stochastic state diagram. With this diagram, we make sure to keep the correctness with modeling validation and simulation for exactly working the embedded system

#### ACKNOWLEDGMENT (*Heading 5*)

This work was supported by the IT R&D Program of MKE/KEIT [10035708, "The Development of CPS (Cyber-Physical Systems) Core Technologies for High Confidential Autonomic Control Software"]

#### REFERENCES

- [1] Miro Samek, Practical Statecharts in C/C++ (Quantum Programming for Embedded Systems), pages 10-20. CMP Books, 2002.
- [2] Jeff Magee, Jeff Kramer, Cuncurrency state models & JAVA Programs, pages 75-114 Wiley, 2004.
- [3] Ho Woo Lee, Queueing Theory, Sigma Press, 2004.



# ICITCS2014

<http://icitcs2014.org/>

ICITCS 5<sup>th</sup> will be held in Beijing, China from October 27th-29th, 2014. This will also include joint conferences from ICISST, ICMWT and ICIEEM. Beijing is a city of great majestic history and has endless activities for anyone to enjoy. We are excited to invite you and your fellow scholars to come join us for a new experience.

# ICISA2015

<http://icisa2015.org/>

Come join us for the 6th ICISA from January 5th-7th, 2015. ICISA2015 will be held in Las Vegas, USA the City of Lights. Las Vegas has everything from amazing views, performances, food and everything for the family. ICISA will have scholars from all over the world and will be the perfect time to come and participate in the conference.

# ICISA 2014

Technically Co-Sponsored by IEEE Computer Society

Technically Co-Sponsored by

