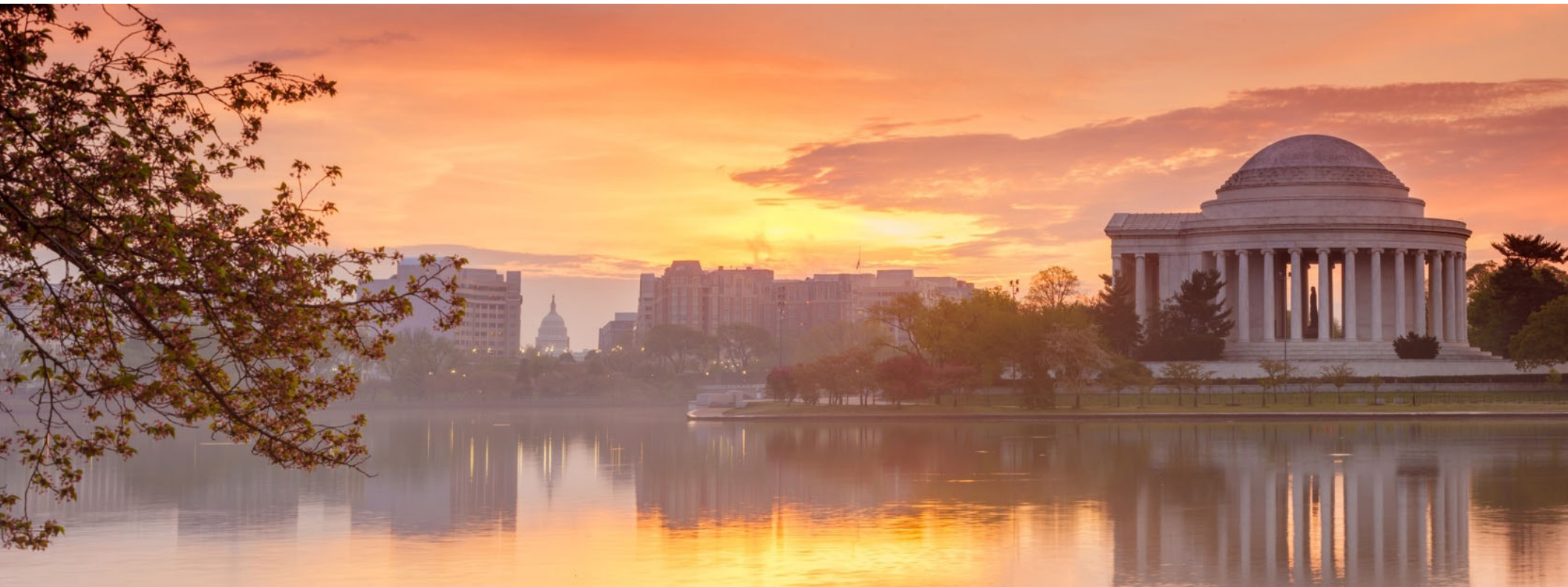


ISSREW 2015

NOVEMBER 2–5, 2015
GAITHERSBURG, MD, USA



The 26th International Symposium on Software Reliability Engineering Workshops



Supplementary Proceedings Table of Contents

Welcome from the General Chairs	i
Message from Industry Chairs	ii
Industrial Track Committees	iii
Doctoral Symposium Welcome Letter	iv
Doctoral Symposium Committees	v
Fast Abstracts Welcome Letter	vi
Fast Abstracts Committees	vii
IWPD Welcome Letter	viii
IWPD Committees	ix
WoSAR Welcome Letter	x
WoSAR Committees	xi
WoSoCer Welcome Letter	xiii
WoSoCer Committees	xiv
BDISW Welcome Letter	xvi
BDISW Committees	xvii
DSW Welcome Letter	xix
WOSD Welcome Letter	xx
WOSD Committees	xxi
Tutorial Abstracts	xxiii

Industrial Sessions:

Session #1: Growth Models

Countdown Graph: Estimating Completion of Testing	1
Rotella, Land, Moshref, and Chulani	
Detection of Unexpected Situations by Applying Software Reliability Growth Model to Test Phases	2
Honda, Washizaki, Fukazawa, Munakata, Morita, Uehara, and Yamamoto	
Resource/Schedule/Content Model: Improving Testing Effectiveness	6
Rotella, Chulani, and Intintolo	

Session #2: Static Analysis and Formal Methods

Static Analysis of Physical Properties in Simulink Models	8
Hocking, Aiello, and Knight	
Test Suites for Benchmarks of Static Analysis Tools	12
Shiraishi, Mohan, and Marimuthu	
Formal Methods for Informal Developers A case-study driven by the French defense agency (DGfasa)	16
Ochem, Perlade	

Session #3: Reliability and availability models and methods

Optimizing Resiliency of Distributed Video Surveillance System for Safer City	17
Machida, Fujiwaka, Koizumi, and Kimura	
Upgrade of the IaaS Cloud: Issues and Potential Solutions in the Context of High-Availability	21
Nabi, Khendek, and Toeroe	
Software-Defined Networking (SDN) Control Message Classification, Verification, and Optimization System	25
Zhu, Song, Park, and Choi	
Automated Generation of Failure Modes and Effects Analysis for a Medical Device	29
Hecht, Nguyen, Chuidian, and Pinchak	

Session #4: Formal Methods and Tools

Integrating Formal Methods with Testing for Reliability Estimation of Component Based Systems	33
Lohar and Dey	
A Fault Injection Description Language for Compiler-based Tools	37
Aliabadi, Pattabiraman, and Bidokhti	
C-SEC (Cyber SCADA Evaluation Capability): Securing Critical Infrastructures	38
Romero-Mariona, Kline	

Session #5: Operational analysis and workflow

Operational Softwarized Networks Reliability Management	39
Song and Zhu	

Knowledge Transition: Discovering Workflow Models from Functional Tests.....	40
Shah, Khadke, Rana	
Effective management of work in a geographically dispersed team using Tasks in Agile methodology	44
Krishnan, Kovvuri, and Balasubramani	

Doctoral Symposium Sessions:

Session #1

An analysis and extension of Category partition testing for constrained systems	55
Sunint Khalsa	
Analyzing failure mechanism for complex software-intensive systems.....	57
Luyi Li	
On Validating UML Consistency Rules.....	59
Damiano Torre	

Session #2

A Scalable and Accurate Hybrid Vulnerability Analysis Framework	61
Julian Thome	
Cloud-based Runtime Verification of Health-caring Systems using Software Agents	63
Negar Majma	

Fast Abstracts Session:

Analyzing the Reduction of Test Suite Redundancy	65
Ingo Pill, Seema Jehan, and Franz Wotawa	
Conditional Slicing: Reducing Dynamic Slices	66
Birgit Hofer, Georg Hinteregger, and Franz Wotawa	
S-CBAC: A Secure Access Control Model Supporting Group Access for Internet of Things.....	67
Bortong Chen, Yu-Lun Huang, and Mesut Güneş	
UML Consistency Rules in Technical Books.....	68
Damiano Torre, Yvan Labiche, Marcela Genero, and Maged Elaasar	
Getting more in less: The power of single/error annotations in category partition.....	69
Sunint Khalsa and Yvan Labiche	

UML Consistency Rules in Technical Books

Damiano Torre^{§,¶}, Yvan Labiche[¶], Marcela Genero[§], Maged Elaasar[¶]

[¶] Carleton University, Department of Systems and Computer Engineering,

Software Quality Engineering Laboratory

1125 Colonel By Drive, Ottawa, ON K1S5B6, Canada

dctorre@sce.carleton.ca, labiche@sce.carleton.ca, marcela.genero@uclm.es, melaasar@gmail.com

[§] University of Castilla-La Mancha, Department of Technologies and Information Systems, ALARCOS Research Group,

Paseo de la Universidad, 4 13071 Ciudad Real, Spain

Abstract—This paper highlights the need of identifying a set of UML consistency rules that is as complete as possible. For that purpose, we propose to complement our previous research works, by searching for such rules directly from technical books on UML-based object-oriented software development.

Keywords— Unified Modeling Language Consistency rules

I. INTRODUCTION

The Unified Modelling Language (UML) [1] is an Object Management Group (OMG) specification that is most frequently used and is the de-facto standard modeling language for object-oriented design and documentation [2]. When UML diagrams convey contradicting or conflicting semantics, the diagrams are said to be inconsistent [3]. Such inconsistencies may be a source of faults in software systems [4]. It is therefore paramount that they get detected (a type of verification activity), analyzed and fixed [5]. Unfortunately no consolidated set of UML consistency rules was available in the literature when we started our work. To identify and validate a set, as complete as possible, of well-accepted consistency rules for UML diagrams, this PhD work is being developed [6] and the following steps have been done: 1) performed a literature review to collect the current state of the art in terms of UML consistency rules [7]; 2) performed a literature review to collect the current state of the art in terms of UML synthesis techniques [8]. This paper proposes to complement our previous work by identifying UML consistency rules in technical textbooks on UML-based object-oriented software development. Indeed, since such books often describe the use of UML for a given software development purpose, following a given methodology, they must, explicitly or not, discuss UML consistency rules.

II. EXAMPLE OF UML CONSISTENCY RULES IN A BOOK

As an example of UML consistency rules discussed in a textbook, we consider Bruegge and Dutoit's chapter 5 (page 189) [9] that introduces the following heuristics for drawing sequence diagrams that are consistent with class diagrams: reading a sequence diagram from left to right, typically, 1) the first lifeline should correspond to the actor who initiated the corresponding use case (another consistency rule asks that each use case be described by a sequence diagram); 2) the second lifeline should be a boundary object; 3) the third lifeline should be the control object that manages the use case; 4) control objects are created by boundary objects

initiating use cases; 5) boundary objects are created by control objects; 6) entity objects are accessed by control and boundary objects; 7) entity objects never access boundary or control objects. Entity objects represent the persistent information tracked by the system, boundary objects represent the interactions between the actors and the system, and control objects are in charge of realizing use cases [9]. These heuristics are presented in another textbook (e.g., [10]) and are not present in the UML standard [1] nor were they identified in our past work on this topic [7].

III. CONCLUSION AND FUTURE WORK

As we commented, technical textbooks on UML-based object-oriented software development contain consistency rules that have not been collected yet. For that reason, our future work will identify UML consistency rules presented in technical books by following a systematic procedure as we previously did [7] for papers published in peer-reviewed journals, international conferences and workshops.

REFERENCES

- [1] OMG: OMG Unified Modeling Language™ - Superstructure Version 2.5. Object Management Group (2013)
- [2] Pender, T.: UML Bible (1st ed.). John Wiley & Sons, Inc., (2003)
- [3] Simmonds, J., Straeten, R.V., Jonkers, V., Mens, T.: Maintaining Consistency between UML Models using Description LogicZ. RSTI – L'Object 10, 231-244 (2004)
- [4] Muskens, J., Bril, R.J., Chaudron, M.R.V.: Generalizing Consistency Checking between Software Views. Proceedings of the 5th Working IEEE/IFIP Conference on Software Architecture, pp. 169-180. (2005)
- [5] Spanoudakis, G., Zisman, A.: Inconsistency management in software engineering: Survey and open research issues. In: Chang, S.K. (ed.) Handbook of Software Engineering and Knowledge Engineering, pp. 329-380. World Scientific Publishing Co., (2001)
- [6] Torre, D.: On collecting and validating UML consistency rules: a research proposal. Doctoral Symposium at EASE 2014. (2014)
- [7] Torre, D., Labiche, Y., Genero, M.: UML consistency rules: a systematic mapping study. EASE 2014. (2014)
- [8] Torre, D., Labiche, Y., Genero, M., Elaasar, M.: UML diagram synthesis techniques: a systematic mapping study. Carleton University (2015), <http://goo.gl/XOGJw4>
- [9] Bruegge, B., Dutoit, A.: Object-Oriented Software Engineering Using UML, Patterns, and Java (3rd ed.). (2009)
- [10] Rosenberg, D., Stephens, M.: Use Case Driven Object Modeling with UML Theory and Practice (2nd Ed.). Apress (2007)