

THE UNIVERSITY OF ADELAIDE
South Australia



THE UNIVERSITY
of ADELAIDE

**Formal Verification of Transactional and Configurable
Service-Oriented Processes**

A dissertation submitted for the degree
Doctor of Philosophy in the School of Computer Science

by

Scott Bourne

Supervised by A/Prof. Michael Sheng and Dr. Claudia Szabo

2016

© Copyright by
Scott Bourne
2016

ABSTRACT OF THE DISSERTATION

Formal Verification of Transactional and Configurable Service-Oriented Processes

by

Scott Bourne

Doctor of Philosophy in School of Computer Science

The University of Adelaide, *South Australia*, 2016

The industrial rise of Web services and cloud services provides ample opportunities for business processes to be implemented with third-party components in a way that is rapid to develop, low-cost, and with reduced start-up risk. *Service-oriented processes* are business processes implemented by remotely provisioning third-party services: autonomous black box implementations of common software or hardware requirements. However, executing a workflow structure of these distributed and heterogeneous components creates several transactional concerns. These include ensuring an acceptable level of atomicity over long-running executions, handling a diverse range of potential fault types, and considering the various transactional properties of component services.

In this thesis, we present three related approaches towards ensuring well-formed transactional behavior in service-oriented processes. We address the problem of identifying issues in the transactional behavior of service-oriented processes at design-time, to prevent costly issues or redevelopment at later stages. We adapt an expressive service-oriented process modeling approach that allows for developers to specify detailed transactional behavior. A set of rules can be applied to this model in order to identify transactional issues such as deadlock and invalid termination. Furthermore, developers can elicit complex and varied transactional requirements for the process with ease using our set of *temporal logic templates*. Model checking is used to ensure that process designs satisfy these rules and requirements.

Recent innovations in *cloud services* have led to the proposal of *Business Process as a Service* (BPaaS). BPaaS offers common business processes as configurable cloud services, enabling clients

to perform complex or resource expensive business operations in a simple pay-be-use manner. Both service providers and clients have concerns to be satisfied during BPaaS configuration. Providers must enforce *domain constraints* to restrict the service to valid configurations, while the client has their own application-dependent requirements for the service to meet. Using *Binary Decision Diagram* (BDD) analysis and model checking as formal methods, we devise a multi-step process that identifies a BPaaS configuration satisfying the requirements of both parties.

These verification and configuration techniques have been implemented in a prototype tool called TL-VIEWS. We include six validation scenarios to demonstrate the effectiveness of our methods, using real Web and cloud services. An extensive performance analysis is performed for each model checking feature used by TL-VIEWS and the results indicate that our state-space reduction measures can decrease verification time for complex models by up to 98.63%.

ORIGINALITY STATEMENT

I certify that this work contains no material which has been accepted for the award of any other degree or diploma in my name, in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text. In addition, I certify that no part of this work will, in the future, be used in a submission in my name, for any other degree or diploma in any university or other tertiary institution without the prior approval of the University of Adelaide and where applicable, any partner institution responsible for the joint-award of this degree.

I give consent to this copy of my thesis, when deposited in the University Library, being made available for loan and photocopying, subject to the provisions of the Copyright Act 1968.

I also give permission for the digital version of my thesis to be made available on the web, via the University's digital research repository, the Library Search and also through web search engines, unless permission has been granted by the University to restrict access for a period of time.

Scott Bourne

SELECTED PUBLICATIONS GENERATED FROM THIS THESIS

1. S. Bourne, C. Szabo, and Q.Z. Sheng. Managing Configurable Business Process as a Service to Satisfy Client Transactional Requirements. In *Proceedings of the 11th International Conference on Services Computing*, IEEE, 2015.
2. S. Bourne, C. Szabo, and Q.Z. Sheng. TL-VIEWS: A Tool for Temporal Logic Verification of Transactional Behavior of Web Service Compositions. In *Proceedings of the 12th International Conference on Service Oriented Computing*, Springer, 2014.
3. S. Bourne, C. Szabo, and Q.Z. Sheng. Verifying Transactional Requirements of Web Service Compositions using Temporal Logic Templates. In *Proceedings of the 14th International Conference on Web Information Systems Engineering*, Springer, 2013.
4. S. Bourne, C. Szabo, and Q.Z. Sheng. Ensuring Well-Formed Conversations Between Control and Operational Behaviors of Web Services. In *Proceedings of the 10th International Conference on Service Oriented Computing*, Springer, 2012.
5. Q. Z. Sheng, Z. Maamar, L. Yao, C. Szabo, and S. Bourne. Behavior Modeling and Automated Verification of Web Services. In *Information Sciences*, Elsevier, 2012.
6. S. Bourne, C. Szabo, and Q.Z. Sheng. Transactional Design Time Verification of Web Service Compositions. (*Under review for Transactions on Services Computing, IEEE*).
7. S. Bourne, C. Szabo, and Q.Z. Sheng. Verification and Assurance of Transactional Requirements in Service-Oriented Processes: A Survey. (*Under review for Computing Surveys, ACM*).

ACKNOWLEDGMENTS

I have been extremely fortunate during my candidature to have the backing of a great support network, both inside and outside my work environment. This thesis simply could not have been completed without all of their presence, input, and encouragement.

I am deeply grateful to my co-supervisor Dr. Claudia Szabo for the time, guidance, effort, input, and knowledge she has dedicated to me over the last few years. Under her encouraging, ambitious, and understanding supervision, I felt I could really produce my best work and grow as a researcher. Her role in improving my ability to *analyze* research work, rather than simply absorb it, cannot be understated.

From day one of my candidature, my principal supervisor Associate Professor Michael Sheng has given me valuable insight into conducting and writing research, while instilling me with the confidence to aim high. I am very appreciative for everything he has done for me over the years, and for the tremendous impact he has had on my professional development.

During my time at the School of Computer Science in The University of Adelaide, there have been many other students that have provided me with support, help, encouragement, and friendship. For all this, I want to thank Yihong, Ali, Javier, Lu, Lachlan, Kewen, Sujith, and Lijuan.

Away from academic life, I am very thankful to my parents for the unwavering support that has been behind every accomplishment I have made. Regular camaraderie from my interstate brother delivered to my inbox has been appreciated, masked in dry humour though it was. I also want to thank my friends Michael, Daniel, Nathan, Diana, Tom, Candice, Angela, and Heather. All the fun we have had together allowed me to keep going.

TABLE OF CONTENTS

1	Introduction	1
1.1	Motivation	1
1.1.1	Reliable Transactional Service-Oriented Processes	2
1.1.2	Transactional Requirements Compliance in Service-Oriented Processes	4
1.1.3	Business Process as a Service Configuration	5
1.2	Goals	6
1.3	Contributions	7
1.4	Thesis Outline	9
2	Background	12
2.1	Advanced Transaction Models	13
2.2	Web Service Composition	16
2.2.1	Transactional Web Service Compositions for Business Processes	19
2.3	Cloud Services	20
2.3.1	Business Process as a Service	23
2.4	Summary	25
3	Well-Formed Transactional Behavior in Service-Oriented Processes	26
3.1	Related Work	27
3.1.1	Comparison Criteria	27
3.1.2	Survey	32
3.1.3	Research Direction	37
3.2	Transactional Service-Oriented Process Modeling	38
3.2.1	Motivating Example	38

3.2.2	Control and Operational Behaviors	40
3.2.3	Inter-Behavior Messages	41
3.3	Well-Formed Inter-Behavior Conversations	43
3.3.1	Conversation Structure Rules	45
3.3.2	Message Sequence Rules	46
3.4	Enabling Formal Verification Through Model Checking	47
3.4.1	Temporal Logic Transformations	47
3.4.2	State Space Reduction	49
3.5	Summary	56
4	Temporal Logic Templates for Application-Dependent Transactional Requirements	58
4.1	Related Work	59
4.1.1	Ensuring Transactional Requirements in Service-Oriented Processes	59
4.1.2	Temporal Logic Patterns	65
4.1.3	Research Direction	67
4.2	Formalizing Transactional Requirements	68
4.2.1	Component-Level Templates	69
4.2.2	Process-Level Templates	72
4.3	Enabling Formal Verification Through Model Checking	74
4.3.1	Transactional Requirement Formalization	74
4.3.2	State Space Reduction	75
4.4	Summary	78
5	Transactional Behavior Verification in Business Process as a Service Configuration	80
5.1	Related Work	81
5.1.1	Comparison Criteria	81

5.1.2	Survey	82
5.1.3	Research Direction	88
5.2	Transactional Business Process as a Service Modeling	89
5.2.1	Motivating Example	90
5.2.2	BPaaS Model	91
5.3	Configuration Domain Constraints	94
5.4	Business Process as a Service Configuration and Verification	95
5.4.1	BPaaS Configuration Process	95
5.4.2	BDD Analysis for Ensuring Domain Constraints	97
5.4.3	Model Checking Against Transactional Requirements	100
5.5	Summary	107
6	Prototype Implementation and Experimental Analysis	109
6.1	Implementation Architecture	110
6.1.1	Verification Against Conversation Rules and Templates	110
6.1.2	BPaaS Configuration	114
6.2	Experimental Analysis	115
6.2.1	Validation Scenarios	116
6.2.2	Performance Analysis	129
6.3	Summary	135
7	Conclusion	137
7.1	Summary	138
7.1.1	Conversation Rule Checking for Well-Formed Transactional Behavior	138
7.1.2	Application-Dependent Transactional Requirement Verification	139
7.1.3	Configuration of Transactional Business Process as a Service	140

7.2	Future Directions	141
7.2.1	Diagnosing Conversation Rule and Transactional Requirement Violations .	141
7.2.2	Preserving Transactional Requirements During Dynamic Configuration . .	143
7.2.3	Business Process as a Service Configuration Framework	144
	References	145
	A Temporal Logic Template Specifications	157
	B Using JDD for BDD Construction	165
	C Checkout Configuration BDD	167
	D Implementation of Online Payment Scenario	169
	E Implementation of Course Enrolment Scenario	173

LIST OF FIGURES

2.1	A Web service composition for sending email or postal mail to a group of customers	18
2.2	Cloud service hierarchy [159]	22
2.3	An abstract example of a BPaaS	24
3.1	Faults of transactional service-oriented processes	30
3.2	Fault-handling of transactional service-oriented processes	31
3.3	Overview of the online payment process	39
3.4	Control and operational behavior models of the online payment composition	41
3.5	Two examples of exchanged inter-behavior messages during successful (a) and failed (b) process execution	44
3.6	Flattened behavior model of the online payment composition	52
3.7	Reduced Kripke structure for verification against conversation rules	55
4.1	Reduced Kripke structure for transactional requirement verification	78
5.1	BPMN model of the configurable checkout BPaaS	92
5.2	Configurable resources and data objects mapped to a BPMN activity	93
5.3	Inter-behavior messages used to enable communication between the checkout BPMN and the control behavior model	94
5.4	Feature model constraints used in our approach	95
5.5	A feature model representing configuration constraints of the checkout BPaaS	96
5.6	Overview of the BPaaS configuration and verification process	97
5.7	A selection of the feature model (a) transformed into propositional logic (b) and a BDD (c)	98

5.8	Binary Decision Diagram form of the feature model in Figure 5.7 with four feature selections	100
5.9	Propositional logic form of the checkout process feature model with the feature selections of Table 5.7	101
5.10	Kripke structure example for verifying the checkout activity selection	103
5.11	An example of how activities with multiple resources are traversed for the second model checking phase	104
5.12	Kripke structure example for verifying resource and data object selection	105
5.13	BPMN of a configuration solution of the Checkout BPaaS	106
6.1	TL-VIEWS architecture	110
6.2	TL-VIEWS process design interface	111
6.3	TL-VIEWS requirement specification window	112
6.4	TL-VIEWS results window following unsuccessful conversation rule verification	117
6.5	Operational behavior model of the lesson enrolment composition	120
6.6	Conversation rule checking output for the online enrolment model	122
6.7	Transactional requirement verification output for the course enrolment model	124
6.8	Violating stack trace produced by NuSMV	127
6.9	Verification times during configuration with and without reduction for 10 to 100 requirements	133
B.1	A Java class for implementing a BDD using the JDD library	166
C.1	Binary Decision Diagram form of the propositional logic property of Figure 5.9	168
D.1	Conversation rules and Kripke structure formulated in an SMV input file for verifying the online payment process	170

D.2	Kripke structure definition and transactional requirements of the online payment process	171
D.3	Kripke structure relation of the online payment process	172
E.1	NuSMV input file containing the temporal logic properties for verifying the course enrolment process against conversation rules	174
E.2	NuSMV input file containing the Kripke structure for verifying the course enrolment process against conversation rules	175
E.3	NuSMV input file containing the course enrolment transactional requirements . . .	176
E.4	NuSMV input file containing the course enrolment Kripke structure relation	177

LIST OF TABLES

3.1	Overview of comparison criteria	28
3.2	<i>Overview</i> criteria of application-independent requirements approaches.	32
3.3	<i>Support</i> criteria of application-independent requirements approaches	33
3.4	<i>Method</i> criteria of application-independent requirements approaches.	34
3.5	<i>Transactional behavior</i> criteria of application-independent requirements approaches.	35
3.6	Initiation inter-behavior messages	42
3.7	Outcome inter-behavior messages	43
3.8	Conversation structure rules	45
3.9	Message sequence conversation rules	46
3.10	Conversation rules formalized using LTL and CTL	48
3.11	A set of inter-behavior messages defined over the online payment design	50
3.12	Temporal logic conversation rules for verifying a Kripke structure	56
4.1	<i>Overview</i> criteria of application-dependent requirements approaches.	61
4.2	<i>Support</i> criteria of application-independent requirements approaches.	62
4.3	<i>Method</i> criteria of application-dependent requirements approaches.	63
4.4	<i>Transactional behavior</i> criteria of application-dependent requirements approaches.	64
4.5	Template specification for <code>CompensateFailure</code>	71
4.6	Template specification for <code>ControlStateCritical</code>	73
4.7	Transactional requirements for the online payment process	75
4.8	Online payment process transactional requirements implemented using our template set	75
4.9	Temporal logic formalizations of the online payment process transactional requirements	78

5.1	Comparison criteria overview	82
5.2	<i>Support</i> criteria for comparing work related to BPaaS configuration	83
5.3	<i>Correctness Criteria</i> for comparing work related to BPaaS configuration	84
5.4	Configurable resources for the checkout BPaaS	93
5.5	Configurable data objects for the checkout BPaaS	93
5.6	Propositional logic representations of the feature model constraints of Figure 5.4	99
5.7	A selection of features from the checkout BPaaS	99
6.1	Transactional requirements for the online payment model	119
6.2	The inter-behavior messages used in the lesson enrolment composition	121
6.3	Transactional requirements for the course enrolment model	123
6.4	Transactional requirements for BPaaS Scenario A	126
6.5	Transactional requirements for BPaaS Scenario B	129
6.6	Additional feature selections for BPaaS Scenario B	129
6.7	Verification time (in seconds) of conversation rules with and without Kripke structure reduction	130
6.8	NuSMV execution times for individual templates	131
6.9	Verification time (in seconds) of temporal logic templates with and without Kripke structure reduction	132
6.10	Details of the configurable BPaaS test suite for performance analysis	134
6.11	Verification time (in seconds) of increasingly complex configuration scenarios	134
A.1	Template specification for <code>CompensateFailure</code>	158
A.2	Template specification for <code>CompensateSuccess</code> , minus temporal logic	158
A.3	LTL for the implementations of <code>CompSuccess</code>	159
A.4	Template specification for <code>Alternative</code>	160

A.5	Template specification for NonRetriable	161
A.6	Template specification for RetriablePivot	161
A.7	Template specification for NonRetriablePivot	162
A.8	Template specification for ControlStateCritical	162
A.9	Template specification for ControlStateTrigger	163
A.10	Template specification for ControlStateReachable	163
A.11	Template specification for ControlStateUnreachable	164
A.12	Template specification for Compensation	164
A.13	Template specification for ConditionalCompensation	164

LIST OF ALGORITHMS

1	Conversation Checking Kripke Structure Reduction: $CKSR$	54
2	Conversation Checking Depth-First Traversal: $CDF(s_x, s_m, S_v)$	54
3	Transactional Requirements Kripke Structure Reduction: $TKSR$	76
4	Transactional Requirements Depth-First Traversal: $TDF(s_c, s_x, s_k, AP_v)$	76