



BUILDING RELIABLE AND ROBUST
SERVICE-BASED SYSTEMS FOR
AUTOMATED BUSINESS PROCESSES

Julian Jang-Jaccard

A thesis submitted in fulfilment of
the requirement for the degree of
Doctor of Philosophy

School of Information Technologies
University of Sydney

March 2007

Abstract

An exciting trend in enterprise computing lies in the integration of applications across an organisation and even between organisations. This allows the provision of services by automated business processes that coordinate business activity among several collaborating organisations. The best successes in this type of integrated distributed system come through use of Web Services and Service-based Architecture, which allow interoperation between applications through open standards based on XML and SOAP. But still, there are unresolved issues when developers seek to build a reliable and robust system.

An important goal for the designers of a loosely coupled distributed system is to maintain consistency for each long running business process in the presence of failures and concurrent activities. Our approach to assist the developers in this domain is to guide the developers with the key principles they must consider, and to provide programming models and protocols, which make it easier to detect and avoid consistency faults in service-based system.

We start by defining a realistic e-procurement scenario to illustrate the common problems faced by the developers which prevent them from building a reliable and robust system. These problems make it hard to maintain the consistency of the data and state during the execution of a business process in the occurrence of failures and interference from concurrent activities. Through the analysis of the common problems, we identify key principles the developers must consider to avoid producing the common problems.

Then based on the key principles, we provide a framework called GAT in the orchestration infrastructure. GAT allows developers to express all the necessary processing to handle deviations including those due to failures and concurrent activities. We discuss the GAT framework in detail with its structure and key features. Using an example taken from part of the e-procurement case study, we illustrate how developers can use the framework to design their business requirements. We also discuss how key features of the new framework help the developers to avoid producing consistency faults. We illustrate how systems based on our framework can be built using today's proven technology.

Finally, we provide a unified isolation mechanism called Promises that is not only applicable to our GAT framework, but also to any applications that run in the service-based world. We discuss the concept, how it works, and how it defines a protocol. We also provide a list of potential implementation techniques. Using some of the implementation techniques we mention, we provide a proof-of-concept prototype system.

Acknowledgements

My deepest thank goes to my supervisor Alan Fekete for his guidance and patience during the course of my doctoral studies at the University of Sydney. His broad knowledge and his clear thinking have always inspired me. Not only for the research aspect, his warm personality and friendly smile have always assured me how lucky I have been to work with someone as capable as he is. Equally, I owe the deepest gratitude to Paul Greenfield. It was Paul who hired me in CSIRO and trained me as a researcher even when I had no idea what it meant. During the last 5 years, with much happening in CSIRO, he has continuously embraced my interests and showed me the right direction: how to be a good researcher and what it means to do quality research. Alan and Paul, I am sure I'll miss very much our regular Friday meetings at 8am.

I would like to thank my colleague Surya Nepal. At many difficult moments during my research career, he has been there for me, listening to my problems and complaints. I have been able to overcome many obstacles through the constructive discussions and friendly advice from him. I also would like to thank Dean Kuo, as without him, this research project would have not started.

I would like thank to my project manager at my work John Zic for providing me an opportunity to undertake my study in conjunction with my work responsibility. I also would like thank to the supervisors and fellow PhD students of the middleware group for their constructive feedback on presentations on my thesis topic. Special thanks go to David Levy, Uwe Roehm, Shiping Chen, and Anna Liu.

This thesis is dedicated to my mom Ok-am Han, my late dad Bong-ok Jang, and my husband Frederic Jaccard. Without the frequent assurance and encouragement from my mom, I would have not made this far with my studies. My husband has been simply superb in showing me continuous love and support. My long journey would have not been the same without you my love. And I am sure that my dad watching over far from there must be feeling proud and happy for his youngest daughter to be where I am today.

Table of Contents

ACKNOWLEDGEMENTS.....	III
INTRODUCTION	1
1.1 Problem Statement	2
1.2 Our Approach.....	3
1.3 Contributions: Understanding the Nature of Service-based System.....	4
1.4 Contributions: GAT – New Event-Driven Programming Model for Defining Business Processes.....	5
1.5 Contributions: Design Principles in Building a Business Process System based on GAT Model.....	5
1.6 Contributions: Promises – New Unified Isolation Mechanisms for Service-based System	6
1.7 Contributions: Design Principles in Supporting Promises.....	8
1.8 Thesis Structure	9
RELATED WORK	10
2.1 Traditional Transaction Support	10
2.1.1 ACID properties.....	10
2.1.2 Locking Mechanism.....	11
2.1.3 Two-Phase Commit (2PC).....	13
2.1.4 Advanced Mechanisms for Standard ACID	15
2.1.5 Extended Transaction Models.....	17
2.2 Distributed Computing Platforms	19
2.2.1 Conventional Middleware.....	20
2.2.2 Workflow Management Systems (WfMS)	22
2.2.3 Business to Business Integration (B2Bi) and Service-oriented Architecture (SOA).....	26
2.3 Summary	34
UNDERSTANDING THE NATURE OF SERVICE-BASED SYSTEMS	35
3.1 Motivating Scenario	35
3.1.1 E-procurement When Ordering Goods	35
3.1.2 Merchant System	36
3.2 Issues for Service-based Systems	38
3.2.1 Time Related Issues	39
3.2.2 No Termination	40
3.2.3 Unprocessed Messages	40

3.2.4 Out of Order Processing of Messages.....	41
3.2.5 Lack of Isolation	42
3.2.6 Cancellations.....	43
3.3 Introduction to Deviations	44
3.3.1 Recoverable Deviations	44
3.3.2 Unrecoverable Deviations.....	45
3.3.3 State-related Deviations	46
3.4 States and State Mismatch	47
3.4.1 States	47
3.4.2 Classification of Deviations	49
3.5 Desired Features in Handling Deviations	54
3.5.1 Cancellations.....	54
3.5.2 Continuing to Make Forward Progress	57
3.6 Critiques of Standard Mechanisms and Supports from Current Technologies.....	59
3.7 Summary	60

**GAT – NEW EVENT-DRIVEN PROGRAMMING MODEL
FOR DEFINING BUSINESS PROCESSES62**

4.1 Payment Process	63
4.1.1 Send Invoice.....	63
4.1.2 Receive Payment.....	64
4.1.3 Send Receipt	66
4.1.4 Cancellations.....	66
4.2 GAT Programming Model.....	68
4.2.1 Structure	68
4.2.2 Key Features	71
4.3 Payment Process in GAT Model.....	72
4.3.1 Activity Group: sendInvoice.....	72
4.3.2 Activity Group: receivePayment.....	74
4.3.3 Activity Group: overduePayment	76
4.3.4 Activity Group: sendReceipt.....	78
4.3.5 Activity Group: cancellations	78
4.4 Experiment with GAT.....	80
4.5 Evaluation Compared to Other Models	81
4.6 Summary	82

**DESIGN PRINCIPLES IN BUILDING A BUSINESS
PROCESS SYSTEM BASED ON GAT MODEL..... 84**

5.1 Case study	84
5.2 GAT Design Consideration.....	86
5.2.1 Control Flow of Business Activities	86
5.2.2 Atomicity/Isolation Issues	87
5.2.3 Management and Distribution of Events	87

5.3 Architecture of GAT Prototype System.....	88
5.3.1 User Interface.....	89
5.3.2 Event Handler.....	89
5.3.3 GAT Processor.....	90
5.3.4 Remote Communication Handler.....	90
5.3.5 Data Storage.....	91
5.3.6 Running Business Systems.....	92
5.3.7 Performance.....	93
5.4 Implementation of GAT model.....	96
5.4.1 Defining Activity Group and Execution of Activities.....	97
5.4.2 Supporting the GAT Event Concept.....	100
5.4.3 Inter-process Communication.....	103
5.5 Design of a General GAT Engine.....	104
5.5.1 GAT Specifications.....	106
5.5.2 Analyser.....	107
5.5.3 Mapper.....	107
5.5.4 CodeDOM.....	107
5.5.5 Generating C#.....	107
5.5.6 Compiling.....	107
5.6 Evaluation Compared to Other Implementation Alternatives.....	108
5.7 Summary.....	110

**PROMISES – NEW UNIFIED ISOLATION MECHANISMS
FOR SERVICE-BASED SYSTEMS..... 112**

6.1 Promises.....	113
6.2 Resources and Predicates.....	116
6.2.1 Anonymous View.....	117
6.2.2 Named View.....	118
6.2.3 View via Properties.....	119
6.3 Atomicity and Promises.....	121
6.4 Implementation Techniques.....	122
6.5 Promise Protocol.....	124
6.6 Promises and Isolation.....	126
6.7 Other Similar Isolation Mechanisms.....	127
6.8 Conclusion.....	129

DESIGN PRINCIPLES IN SUPPORTING PROMISES..... 130

7.1 Design Issues and Constraints of Promises.....	130
7.1.1 Compatibility.....	131
7.1.2 Representing Promises.....	131
7.1.3 Promises and Schemas.....	131
7.1.4 Isolation and Concurrency.....	132
7.1.5 Dynamic Promise List.....	132
7.2 Structure.....	133
7.2.1 Messages.....	134

7.2.2 Components	135
7.2.3 Promise Consistency Checking.....	137
7.2.4 Promise Operations.....	138
7.3 Reflecting on our design	139
7.3.1 Compatibility	139
7.3.2 Representing Promises.....	139
7.3.3 Promises and Schemas.....	140
7.3.4 Isolation and Concurrency	140
7.3.5 Dynamic Promise List.....	141
7.4 Implementation	142
7.4.1 Overview of Promise Consistency Checking Interface	142
7.4.2 Implementation of Promise Operations	146
7.5 Other Alternatives	155
7.6 Summary	156
CONCLUSIONS	158

List of Figures

Figure 1 Conventional Locking Example (source from [25])	12
Figure 2 Two-Phase Commit (2PC) Protocol.....	14
Figure 3 Escrow Locking Example (source from [25])	16
Figure 4 Long Transaction in Saga.....	18
Figure 5 Conventional Workflow Example.....	22
Figure 6 ECA Example.....	23
Figure 7 E-procurement Scenario	36
Figure 8 Merchant Workflow	37
Figure 9 Payment Process within the Merchant Process	63
Figure 10 Receive Payment	65
Figure 11 GAT Process Structure.....	69
Figure 12 GAT Activity Structure	70
Figure 13 Activity Group: receivePayment in GAT.....	86
Figure 14 Architecture of GAT Prototype System	88
Figure 15 Snapshot of the Running Prototype System	93
Figure 16 Performance at Each Business Activity in Milliseconds (ms)	94
Figure 17 Snapshot of the Performance Monitor.....	96
Figure 18 GAT Engine Concepts.....	105
Figure 19 GAT Engine Major Stages	106
Figure 20 Example of GAT Syntax	106
Figure 21 Outline of Ordering Process Code.....	127
Figure 22 Structure of Promise System	134
Figure 23 Promise Manager Flow Chart.....	136
Figure 24 Message Sequence on Making New Promises	147
Figure 25 Message Sequences on Executing Actions.....	151
Figure 26 Message Sequence on Updating Promises	154

Publications on which this Thesis is Based

- **Jang, J.**, Fekete, A., Greenfield, P. Delivering Promises for Web Services Applications. Technical Report of University of Sydney School of Information Technologies, TR-605 December 2006.
- Greenfield, P., Fekete, A., **Jang, J.**, Kuo, D., Nepal, S. Isolation Support for Service-based Applications. *In Proceedings of the 3rd biennial Conference on Innovative Data Systems Research (CIDR)*, pp 314-323, Asilomar, USA, January 2007.
- **Jang, J.**, Fekete, A., Nepal, S., Greenfield, P. An Event-Driven Workflow Engine for Service-based Business System. *In Proceedings of the 10th IEEE International Conference on Enterprise Computing (EDOC)*, pp 233-242, Hong Kong, China, October 2006.
- Nepal, S., Fekete, A., Greenfield, P., **Jang, J.**, Kuo, D., Shi, T. A Service oriented Workflow Language for Robust Interacting Applications. *In Proceedings of the 13th Cooperative Information Systems (CoopIS)*, pp 40-58, Cyprus, November 2005.
- Kuo, D., Fekete, A., Greenfield, P., **Jang, J.** Just What Could Possibly Go Wrong In B2B Integration? *In Proceedings of the 27th Annual International Computer Software and Applications Conference (COMSAC)*, pp 544-549, Dallas, USA, November 2003.
- Greenfield, P., Fekete, A., **Jang, J.**, Kuo, D. What are the consistency requirements for B2B systems? *High Performance Transactions Systems (HPTS) Workshop*, Asilomar, California, USA, October 2003.
- Greenfield, P., Fekete, A., **Jang, J.**, Kuo, D. Compensation is Not Enough. *In Proceedings of the 7th IEEE International Enterprise Distributed Object Computing Conference (EDOC)*, pp 232-239, Brisbane, Australia, September 2003.
- **Jang, J.**, Fekete, A., Greenfield, P., Kuo, D. Expressiveness of Workflow Description Languages. *In Proceedings of the 1st International Conference on Web Services (ICWS)*, pp 104-110, Las Vegas, USA, June 2003.

- Fekete, A., Greenfield, P., Kuo, D., **Jang, J.** Transactions in Loosely Coupled Distributed Systems. *In Proceedings of the 14th Australasian Database Conference (ADC)*, pp 7-12, Adelaide, Australia, February 2003.
- Kuo, D., Fekete, A., Greenfield, P., **Jang, J.** Towards a Framework for Capturing Transactional Requirements of Real Workflows. *The 2nd International Workshop on Cooperative Internet Computing (CIC)*, Hong Kong, August 2002.