Models and Verification of BPEL

Franck van Breugel[1] and Maria Koshkina[2] *

[1] York University
4700 Keele Street, Toronto, M3J 1P3, Canada
franck@cs.yorku.ca
[2] IBM
8200 Warden Avenue, Markham, L6G 1C7, Canada
mkoshkin@ca.ibm.com

Abstract. The Web Services Business Process Execution Language (BPEL for short) is a recently developed language that is used to specify compositions of web services. In the last few years, a considerable amount of work has been done on modelling (parts of) BPEL and developing verification techniques and tools for BPEL. In this paper, we provide an overview of the different models of BPEL that have been proposed. Furthermore, we discuss the verification techniques for BPEL that have been put forward and the verification tools for BPEL that have been developed.

Introduction

The Business Process Execution Language for Web Services (BPEL4WS or BPEL for short) was proposed by BEA, IBM and Microsoft. In July 2002, the first version of BPEL was published [27]. Subsequently, SAP and Siebel joined the effort. The second version of BPEL [7] was published in May 2003. That same month, BEA, IBM, Microsoft, SAP and Siebel submitted BPEL to the Organization for the Advancement of Structured Information Standards (OASIS for short) for standardization purposes and the Web Services Business Process Execution Language Technical Committee (WSBPEL TC, for short) was formed. Since then, many major vendors have joined the WSBPEL TC, including Adobe, Hewlett-Packard, NEC, Oracle and Sun. The language has been renamed to the Web Services Business Process Execution Language (WS-BPEL or BPEL for short). The latest version of BPEL can be found in [10].

BPEL represents a convergence of two languages: the Web Services Flow Language (WSFL) [72] of IBM and XLANG [111] of Microsoft. WFSL, XLANG and BPEL are languages to compose web services. Numerous introductions to BPEL can be found on the web and in the literature, including, for example, [28]. Even the first books about BPEL have already appeared (see, for example, [66]). For a detailed comparison of the languages BPEL, WSFL and XLANG we refer the reader to, for example, [4].

* This research was supported by IBM and NSERC.

Like most languages, (the semantics of) BPEL is defined in English prose (see [7, 10, 27]). Such descriptions, although often masterpieces of apparent clarity, usually suffer from inconsistency, ambiguity and incompleteness. Also the (initial) definition of BPEL suffered from inconsistencies (see, for example, [26, Issue 39]), ambiguities (see, for example, [26, Issue 111]) and incompleteness (see, for example, [26, Issue 32]). The WSBPEL TC recognized the need for a formalism to define (the semantics of) BPEL (see [26, Issue 42]). Formalizing the definition of BPEL eliminates inconsistencies and ambiguities and provides a complete description of the language. Such a formal definition may prove fruitful when implementing BPEL, when developing BPEL processes and when reasoning about those processes. For a detailed discussion of the merits of formally defined models we refer the reader to, for example, [96]. Different formalisms have been exploited to formally define (the semantics of) BPEL. In this paper, we will present an overview of the various models that have been developed for BPEL.

Due to the presence of concurrency and intricate features like compensation handling, correlation and death-path-elimination, BPEL processes are error-prone. In addition, BPEL processes may use valuable resources in the form of invocations of web services. Therefore, there is a need to ensure that BPEL processes behave correctly. Testing is an effective way to detect incorrect behaviour. Often it is beneficial to also exploit verification techniques and tools to detect incorrect behaviour. For a detailed discussion of the benefits of verification we refer the reader to, for example, [13]. The need for verification of business processes, like those expressed in BPEL, is argued in, for example, [69]. Different verification techniques and tools have been developed for BPEL. In this paper, we will present an overview of these techniques and tools.

Research on modelling and verifying BPEL processes has been published in the proceedings of numerous conferences and workshops and in several journals. In particular, the International Conference on Business Process Management (BPM) [6, 29, 5], the International Conference on Web Services (ICWS) [124, 1, 2] and the Workshop on Web Services and Formal Methods (WS-FM) [19, 18] are popular venues to present this type of research. There are a few papers that present an overview of this research area (see, for example, [63, 64]). However, these overviews are not as focused and extensive as the one we present here. In the rest of this paper, we will discuss almost 90 papers on models and verification of BPEL.

Acknowledgements

The authors would like to thank Andrew Rimes and Christian Stahl for their help with the literature search.

1 Petri Nets

Petri nets are a formal model for concurrency. A Petri net is a directed, connected, and bipartite graph in which each node is either a place or a transition.

Tokens occupy places. When there is at least one token in every place connected to a transition, the transition is enabled. Any enabled transition may fire removing one token from every input place, and depositing one token in each output place. For an introduction to Petri nets refer the reader to, for example, [102].

Petri nets have been extensively used to model and verify business processes. For an overview, we refer the reader to, for example, [3].

Since the semantics of Petri nets is formally defined, by mapping each BPEL process to a Petri net a formal model of BPEL can be obtained. Not only does this approach provide a model. It also allows the verification techniques and tools developed for Petri nets to be exploited in the context of BPEL processes. This approach has been taken by several research groups.

1.1 The German School

In [109], Schmidt and Stahl discuss a mapping from BPEL to Petri nets by giving several examples. Each BPEL construct is mapped into a Petri net pattern. The complete transformation from BPEL to Petri nets is given by Stahl in [110]. Hinz, Schmidt and Stahl [61] describe the tool BPEL2PN that implements the transformation when abstracting from data. The details of this tool are presented by Hinz in [60]. As shown in [61], the resulting Petri net can be verified using the tool LoLA [108]. LoLA, which stands for a Low Level Analyzer, supports the verification of standard properties of Petri nets, like, for example, determining if a Petri net contains a deadlock, and the verification of properties expressed in the logic CTL.

In most of his work [75–82], Martens focuses on Petri nets rather than BPEL processes. However, since there is a mapping from BPEL processes to Petri nets, all his results are directly applicable to BPEL. Martens introduces several criteria for business processes and their compositions. Next, we will roughly capture them in terms of BPEL. A BPEL process is called usable (or controllable) if there exists an environment with which the process can interact such that the process terminates properly. Two BPEL processes are called compatible if their composition is usable. A BPEL is said to simulate another BPEL process if each environment that makes the latter usable makes the former usable as well. Two BPEL processes are called equivalent (or consistent) if the one simulates the other and vice versa. Martens also presents algorithms to check if BPEL processes satisfy these criteria. These algorithms have been implemented in the tool WOMBAT [83, 84].

In [107], Schlingloff, Martens and Schmidt also consider the usability problem. They show that usability can be expressed in alternating-time temporal logic. As a consequence, model checking algorithms for this logic can be exploited to check for usability. Reisig, Schmidt and Stahl [104] and Lohmann, Massuthe, Stahl and Weinberg [73] also consider the usability (or controllability) problem.

In [103], Reisig proposes to model BPEL by means of a special type of Petri nets called business process nets.

1.2 The Business Process Management Center

In [113], Verbeek and van der Aalst focus on the structured activities of BPEL. They present a mapping of these structured activities to a class of Petri nets called workflow nets. For this class of Petri nets, a verification tool named Wolfan [114] has been developed. This tool can verify properties like, for example, termination of a workflow net and detection of nodes that can never be activated. In their mapping from BPEL to workflow nets, they also consider links, join conditions and dead-path-elimination.

In [95], Ouyang, van der Aalst, Breutel, Dumas, ter Hofstede and Verbeek provide a mapping of all control-flow constructs of BPEL into Petri nets. As a consequence, the authors provide a formal semantics of BPEL. In [94, 95], Ouyang et al. describe two tools that, if used in combination, allow for automated verification of BPEL processes. The BPEL2PNML tool is used to perform a translation from BPEL into the Petri Net Modeling Language (PNML). The resulting model is used as input for the WofBPEL tool. The latter tool has been built using the earlier mentioned Woflan tool. The following three analyses have been implemented in WofBPEL: detection of unreachable activities, detection of multiple simultaneously enabled activities that may consume the same type of message, and determination, for each possible state of a process execution, which types of messages may be consumed in the rest of the execution.

1.3 Coloured Petri Nets

Yang, Tan, Xiao, Yu and Liu [117–121] consider coloured Petri nets. They use coloured Petri nets as these provide a more compact way to model business processes than ordinary Petri nets. Yang et al. show how to map most of the basic and structured activities of BPEL and the Web Service Choreography Interface (WSCI), another language to describe web service composition, to coloured Petri nets.

Also Yi and Kochut [122, 123] focus on coloured Petri nets. They sketch how to verify BPEL processes. Furthermore, they show how the skeleton of a BPEL process can be generated from a coloured Petri net.

2 SPIN

SPIN is a tool to verify software systems. It accepts programs written in the process meta language (Promela) and properties specified in linear temporal logic (LTL) as input. Provided that the Promela program is bounded, SPIN can check if the program satisfies the LTL property. For more details about SPIN, we refer the reader to, for example, [62].

A wide variety of software systems have been expressed in Promela and many interesting properties can be captured in LTL. In order to exploit SPIN to verify BPEL processes, one has to translate BPEL into Promela. A number of researchers have developed translations of (a part of) BPEL into Promela. Below, we will discuss their work.

Since Promela has a formally defined semantics (see, for example, [62, Chapter 7]), a map from BPEL to Promela provides us with a formal model of BPEL.

2.1 The Santa Barbara Group

Fu, Bultan and Su [51, 54] present a framework to verify properties of a web service composition consisting of multiple BPEL processes that communicate asynchronously. Each BPEL process is translated to a guarded automaton. Subsequently, these guarded automata are mapped to Promela. The combination of these translations provide a map from (a part of) BPEL to Promela. Such a two step approach allows for the support of other languages than BPEL and Promela in the future. Furthermore, Fu et al. put forward sufficient conditions so that asynchronous communication can be replaced with synchronous communication without changing the semantics. This replacement simplifies the verification problem.

To handle XML data and XPath expressions, Fu et al. [52] show how these can be expressed in Promela. The Model Schema Language (MSL) is a formal model of XML Schema. MSL is mapped to Promela. Also XPath expressions are translated into Promela. In this way, also data manipulation in BPEL processes can be mapped to Promela and, hence, can be verified in SPIN.

In [53], Fu et al. present the Web Service Analysis Tool (WSAT). This tool contains, for example, a translator of BPEL processes to guarded automata. [20, 21] provide overviews of the work described in this section. For more details, we refer the reader to the thesis of Fu [50].

2.2 Nakajima

In [89, 90], Nakajima presents a translation of the WSFL activities—WSFL is a predecessor of BPEL—into Promela and, hence, a formal model of the WSFL activities. Furthermore, Nakajima shows how SPIN can be exploited to verify web service compositions expressed in WSFL. Nakajima considers BPEL in [92, 93]. The translation of BPEL activities into Promela is split into two parts. First, a BPEL activity is mapped to an extended finite automaton. This provides a formal model for BPEL activities. Second, the automaton is represented in Promela. In the translation abstraction techniques are exploited. This allows for more precise results.

In order to study information leakage in web service compositions, Nakajima [91] proposes to decorate BPEL processes with security labels. Such a decorated BPEL activity can be translated into Promela. Also the environment with which the activity interacts is represented in Promela. Nakajima shows how SPIN can be exploited to detect information leakage.

2.3 Other Approaches

Arias-Fisteus, Fernández and Kloos [8, 9] introduced a tool called VERBUS, standing for verification for business processes. This tool has been developed in a modular way so that it can support multiple languages to compose web services and so that it can exploit multiple model checkers. Currently, VERBUS supports BPEL and the model checkers SPIN, SMV [86] and NuSMV [23]. The tool implements a translation of most BPEL activities to finite state machines. These finite state machines are subsequently mapped onto Promela and the input language for SMV and NuSMV.

3 Process Algebras

A process algebra is a rather small concurrent language that abstracts from many details and focuses on particular features. Numerous process algebras have been introduced including, for example, the Calculus of Communicating Systems (CCS) [87], LOTOS [15] and the π-calculus [88]. Process algebras are usually modelled by means of labelled transition systems. The transition relation of the labelled transition system is generally defined by a collection of axioms and rules. Many different equivalence relations on the set of states of the labelled transition system have been introduced. These behavioural equivalences capture which states behave the same. For an overview of the work on process algebra, we refer the reader to, for example, [14].

Bordeaux and Salaün present an overview of the applicability of process algebras in the context of web services in [16].

Existing process algebras and also new process algebras have been used to model BPEL. Below, we give an overview of this work.

3.1 Labelled Transition System Analyzer

In [74], Kramer and Magee present a process algebra named FSP (Finite State Process). Each FSP represents a finite labelled transition system. The formal model for FSP can be found in [74, Appendix C]. Kramer and Magee also present a tool for FSP named Labelled Transition System Analyzer (LTSA). This tool takes as input an FSP and translates it into a labelled transition system. Subsequently, this labelled transition is analyzed. LTSA can check for safety and progress properties as well as properties expressed in the logic LTL.

Foster, Uchitel, Magee, and Kramer have developed an extension of LTSA to verify BPEL processes. This tool is named LTSA-WS. A key component of the tool is the translation of the activities of BPEL into FSPs. A detailed description of this translation can be found in [42, Appendix C]. Since FSP has a formal semantics, this translation provides a formal model for part of BPEL.

In [44], Foster et al. show how LTSA can be exploited to check if a web service composition implemented in BPEL satisfies a web service composition specification captured by Message Sequence Charts (MSCs). Both the BPEL process and the MSC are translated into FSPs. In [43], Foster et al. use LTSA to check compatibility of web service compositions in BPEL. A case study by Foster, Uchitel, Magee, Kramer and Hu is presented in [49].

Foster, Uchitel, Magee, and Kramer [46] extended their earlier work to model and verify multiple interacting BPEL processes. The tool LTSA-WS was reimplemented as an Eclipse plug-in [45]. An overview of this work can be found in [41, 47].

In [48], Foster et al. extend their framework by also considering the Web Service Choreography Description Language (WS-CDL). In this language one can describe how web services should interact. Foster et al. present a translation between WS-CDL and FSP. Given multiple BPEL processes and a WS-CDL specification, the extension of the LTSA-WS tool translates all into FSPs and checks if the BPEL processes and the WS-CDL specification are consistent.

3.2 Concurrency Workbench

Salaün, Bordeaux and Schaerf [105] advocate to use existing process algebras to model web service compositions like those expressed in BPEL. In particular, they show how the process algebra CCS can be exploited. The Concurrency Workbench (CWB) tool [24] can subsequently be used to check if the resulting CCS processes are behaviourally equivalent or if a CCS process satisfies a property expressed in a logic like CTL*.

The authors [70, 71] introduce a process algebra named the BPE-calculus to model most activities of BPEL. The focus is on links and dead-path-elimination. Given the syntax of the BPE-calculus, in terms of a grammar, and the semantics of the BPE-calculus, in terms of a collection of axioms and rules, the Process Algebra Compiler [25] generates a module. This module can be incorporated into the CWB, resulting in a tool that can also handle BPE-processes. This tool can verify properties of BPE-processes. In [65], Huynh presents a mapping from BPEL processes to BPE-processes. This mapping allows us to verify BPEL processes by means of the extended CWB.

3.3 LOTOS

In [40, 106], Ferrara, Salaün and Chirichiello present a two-way mapping between the process algebra LOTOS and BPEL. Most BPEL activities including fault handlers, compensation handlers and event handlers are considered. By going from BPEL to LOTOS, the toolbox CADP [39], standing for Construction and Analysis of Distributed Processes, can be exploited for the verification of BPEL processes. Counterexamples produced by CADP, given in LOTOS, are mapped back to BPEL.

In [112], CADP is also proposed as the basis of a tool for the verification of BPEL processes. Tremblay and Chae suggest to translate a BPEL activity to a LOTOS process and to map its specification, expressed as a path expression, to a mu-calculus expression. Subsequently, CADP can be exploited to verify if a BPEL process conforms to its specification.

3.4 Other Approaches

In [99], Pu, Zhao, Wang and Qiu introduce a process algebra that is based on the activities of BPEL and focuses on fault and compensation handling. Pu et al. provide a formal model for their process algebra. In [100], Pu, Zhu, Qiu, Wang, Zhao and He extend the process algebra. For example, the switch and while activity and links are covered. Pu et al. extend the model to deal with the additional constructs. Furthermore, they introduce a behavioural equivalence which relates those processes that behave the same. The process algebra, this time without compensation handling, is considered in [101]. A translation that maps each process to a network of timed automata is presented. The translation has been proved correct and it has been implemented. The resulting network of timed automata can be used as input for the UPPAAL tool [12]. This tool can check properties expressed in a subset of the logic CTL.

In [59] and [58, Chapter 3], Hamadi and Benatallah introduce a process algebra to model web service composition. They do not focus on BPEL in particular, but most basic and structured activities of BPEL can easily be expressed in their process algebra. To provide a semantics for the process algebra, each process is mapped to a Petri net.

Butler, Ferreira and Ng [22] model almost all BPEL activities by mapping them to the process algebra StAC, which stands for Structured Activity Compensation, enriched with the B notation. The B notation is exploited to handle data. The focus is on compensation handlers. Since the semantics of StAC is formally defined, this provides us with a model for part of BPEL.

Mazzara and Lucchi [85] extend the π-calculus with event notification, by adding two new constructs: one to notify an event and another to associate a scope with an event. The semantics of this extended calculus is defined in terms of a labelled transition system. Mazzara and Lucchi show that BPEL's exception handling, event handling and compensation handling can be expressed in the calculus.

Viroli [115] presents a process algebra that captures most BPEL activities and focuses on correlation. The process algebra is modelled by means of a labelled transition system.

4 Abstract State Machines

Abstract state machines (ASMs) have been used to model a large variety of languages. A basic ASM consists of a finite set of transition rules. Each transition rule consists of two parts: a Boolean expression and a finite set of assignments. The transition rules captures which transitions the ASM can make. A transition takes the ASM from one state to another. The latter state is obtained from the former state by performing the assignments of those transition rules whose Boolean expressions evaluate to true. For an introduction to the ASM approach, we refer the reader to, for example, [17].

ASMs have also been used to model BPEL. Below, we provided a brief overview of this work.

4.1 The SFU Group

A group at Simon Frasier Univerisity has provided a semantic model for BPEL using the ASM approach. Farahbod, Glässer and Vajihollahi [34–38] model all key aspects of BPEL. For example, the basic and structured activities, correlation, and compensation, event and fault handling are modelled. To model interaction, Farahbod et al. introduce so-called inbox and outbox managers that deal with the message exchanges. For dealing with some of real time aspects of BPEL, like time-outs, an abstract notion of global system time is introduced and additional constraints on the sequences of transition are imposed.

4.2 Fahland and Reisig

The ASM model for BPEL proposed by Fahland and Reisig [32, 33, 103] extends and refines the SFU model. Reisig discusses the model by means of an example in [103]. In [33], the focus is on fault handlers and event handlers. It is shown how these BPEL features can be modelled within the ASM framework. [32] can be viewed as a variation on and an extension of the model developed by the SFU group. For example, Fahland models dead-path-elimination. [32] provides a complete model of BPEL.

5 Automata

Wombacher, Fankhauser and Neuhold [116] present a translation of most BPEL activities into annotated deterministic finite automata. The states of the automata are annotated with Boolean expressions. These Boolean expressions capture how a BPEL process interacts with its environment. Since deterministic finite automata have a well-defined semantics, the transformation provides a model for most BPEL activities.

In [57], Haddad, Melliti, Moreaux and Rampacek model some of the activities of XLANG, one of BPEL's predecessors, by means of labelled transition systems. The transitions capture the passing of time in a discrete way. Furthermore, Haddad et al. define when two labelled transition systems, modelling XLANG processes, interact correctly. In [56], Haddad et al. extend their results from discrete time to real time. Instead of labelled transition systems, timed automata are used to model the XLANG activities.

In [67], Kazhamiakin and Pistore focus on three communication models of business processes: synchronous, ordered asynchronous, and unordered asynchronous. Given a number of communicating BPEL processes, each process is transformed into a state transition system and subsequently these systems are composed in parallel, resulting in yet another state transition system. The resulting system can be fed into the NuSMV tool to check the validity of the system with respect to a given communication model. Furthermore, NuSMV can also be exploited to verify properties of the system.

In [98], Pistore, Traverso, Bertoli and Marconi present a number of tools for BPEL. The tool BPEL2STS translates an (abstract) BPEL process to a state transition system. A number of these state transition systems, representing BPEL processes, are composed in parallel. The resulting parallel composition is also represented by a state transition system. The tool MBP takes as input such a parallel composition and a requirement, the latter being formalized in EaGLe. As output, MBP produces a state transition system such that this system in parallel with the input system satisfies the input specification. The tool STS2BPEL translates a state transition system to a BPEL process. Combined these tools can synthesize web service compositions expressed in BPEL.

Baldoni, Baroglio, Martelli, Patti and Schifanella [11] propose a formal framework that can be applied for checking conformance of an implementation, as described in, for example, BPEL, to a specification, as described in, for example, WS-CDL, and for checking if two implementations, as described in, for example, BPEL, are compatible. Both the BPEL process and the WS-CDL specification are mapped to a deterministic finite automaton.

6 Other Models and Verification Techniques and Tools

Duan, Bernstein, Lewis and Lu [31, 30] present a weakest precondition and a strongest postcondition semantics for some of the BPEL activities. Also an axiomatic semantics for these activities is given. Furthermore, Duan et al. have implemented a tool that annotates activities with pre- and postconditions.

In [97], Pistore, Rovera and Busetta use Formal Tropos (FT) [55] to specify business processes. By means of a set of formal techniques, a BPEL process is extracted from an FT specification. Pistore et al. have extended the T-Tool [55] with a translation of BPEL activities to finite state machines. These finite state machines can be used as input to the tool NuSMV, which can subsequently be exploited to verify properties of the finite state machines and, hence, of the BPEL activities. In [68], Kazhamiakin, Pistore and Roveri show how an FT specification of a business process can be encoded in Promela, the input language of SPIN.

References

1. *Proceedings of the 2nd IEEE International Conference on Web Services*, San Diego, CA, USA, July 2004. IEEE.
2. *Proceedings of the 2005 IEEE International Conference on Web Services*, Orlando, FL, USA, July 2005. IEEE.
3. W.M.P. van der Aalst. Challenges in business process management: Verification of business processes using Petri nets. *Bulletin of the EATCS*, 80:174–199, June 2003.
4. W.M.P. van der Aalst. Don't go with the flow: Web services composition standards exposed. *IEEE Intelligent Systems*, 18(1):72–76, January/February 2003.
5. W.M.P. van der Aalst, B. Benatallah, F. Casati, and F. Curbera, editors. *Proceedings of the 3rd International Conference on Business Process Management*, volume 3649 of *Lecture Notes in Computer Science*, Nancy, France, September 2005. Springer-Verlag.

6. W.M.P. van der Aalst, A.H.M. ter Hofstede, and M. Weske, editors. *Proceedings of the International Conference on Business Process Management*, volume 2678 of *Lecture Notes in Computer Science*, Eindhoven, The Netherlands, June 2003. Springer-Verlag.
7. T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, and S. Weerawarana. Business process execution language for web services, version 1.1, May 2003.
8. J. Arias-Fisteus, L.S. Fernández, and C.D. Kloos. Formal verification of BPEL4WS business collaborations. In K. Bauknecht, M. Bichler, and B. Pröll, editors, *Proceedings of the 5th International Conference on Electronic Commerce and Web Technologies*, volume 3182 of *Lecture Notes in Computer Science*, pages 76–85, Zaragoza, Spain, August/September 2004. Springer-Verlag.
9. J. Arias-Fisteus, L.S. Fernández, and C.D. Kloos. Applying model checking to BPEL4WS business collaborations. In H. Haddad, L.M. Liebrock, A. Omicini, and R.L. Wainwright, editors, *Proceedings of the 2005 ACM Symposium on Applied Computing*, pages 826–830, Santa Fe, NM, USA, March 2005. ACM.
10. A. Arkin, S. Askary, B. Bloch, F. Curbera, Y. Goland, N. Kartha, C.K. Liu, V. Mehta, S. Thatte, P. Yendluri, A. Yiu, and A. Alves. Web services business process execution language, version 2.0, December 2005.
11. M. Baldoni, C. Baroglio, A. Martelli, V. Patti, and C. Schifanella. Verifying the conformance of web services to global interaction protocols: a first step. In M. Bravetti, L. Kloul, and G. Zavattaro, editors, *Proceedings of the 2nd International Workshop on Web Services and Formal Methods*, volume 3670 of *Lecture Notes in Computer Science*, pages 257–271, Versailles, France, September 2005. Springer-Verlag.
12. G. Behrmann, A. David, and K.G. Larsen. A tutorial on UPPAAL. In M. Bernardo and F. Corradini, editors, *Proceedings on the International School on Formal Methods for the Design of Computer, Communication, and Software Systems*, volume 3185 of *Lecture Notes in Computer Science*, pages 200–236, Bertinora, Italy, September 2004. Springer-Verlag.
13. B. Bérard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, and Ph. Schnoebelen. *Systems and Software Verification*. Springer-Verlag, Berlin, Germany, 2001.
14. J.A. Bergstra, A. Ponse, and S.A. Smolka, editors. *Handbook of Process Algebra*. Elsevier, Amsterdam, The Netherlands, 2001.
15. T. Bolognesi and E. Brinksma. Introduction to the ISO specification language LOTOS. *Computer Networks*, 14:25–59, 1987.
16. L. Bordeaux and G. Salaün. Using process algebra for web services: Early results and perspectives. In M. Shan, U. Dayal, and M. Hsu, editors, *Proceedings of the 5th International Workshop on Technologies for E-Services*, volume 3324 of *Lecture Notes in Computer Science*, pages 54–68, Toronto, Canada, August 2004. Springer-Verlag.
17. E. Börger and R. Stärk. *Abstract State Machines: A Method for High-Level System Design and Analysis*. Springer-Verlag, New York, NY, USA, 2003.
18. M. Bravetti, L. Kloul, and G. Zavattaro, editors. *Proceedings of the 2nd International Workshop on Web Services and Formal Methods*, volume 3670 of *Lecture Notes in Computer Science*, Versailles, France, September 2005. Sringer-Verlag.
19. M. Bravetti and G. Zavattaro, editors. *Proceedings of the 1st International Workshop on Web Services and Formal Methods*, volume 105 of *Electronic Notes in Theoretical Computer Science*, Pisa, Italy, February 2004. Elsevier.

20. T. Bultan, X. Fu, and J. Su. Tools for automated verification of web services. In F. Wang, editor, *Proceedings of the 2nd International Conference on Automated Technology for Verification and Analysis*, volume 3299 of *Lecture Notes in Computer Science*, pages 8–10, Taipei, Taiwan, October/November 2004. Springer-Verlag.
21. T. Bultan, X. Fu, and J. Su. Analyzing conversations of web services. *IEEE Internet Computing*, 10(1):18–25, January/February 2006.
22. M. Butler, C. Ferreira, and M.Y. Ng. Precise modelling of compensating business transactions and its application to BPEL. *Journal of Universal Computer Science*, 11(5):712–743, May 2005.
23. A. Cimatti, E.M. Clarke, F. Giunchiglia, and M. Roveri. NuSMV: a new symbolic model verifier. *Software Tools for Technology Transfer*, 2(4):410–425, March 2000.
24. R. Cleaveland and S. Sims. The NCSU concurrency workbench. In R. Alur and T. Henzinger, editors, *Proceedings of the 8th International Conference on Computer Aided Verification*, volume 1102 of *Lecture Notes in Computer Science*, pages 394–397, New Brunswick, NJ, USA, July 1996. Springer-Verlag.
25. R. Cleaveland and S.T. Sims. Generic tools for verifying concurrent systems. *Science of Computer Programming*, 42(1):39–47, January 2002.
26. Web Services Business Process Execution Language Technical Committee. WS BPEL issues list, April 2006.
27. F. Curbera, Y. Goland, J. Klein, F. Leymann, D. Roller, S. Thatte, and S. Weerawarana. Business process execution language for web services, version 1.0, July 2002.
28. F. Curbera, R. Khalaf, N. Mukhi, S. Tai, and S. Weerawarana. The next step in web services. *Communications of the ACM*, 46(10):29–34, October 2003.
29. J. Desel, B. Pernici, and M. Weske, editors. *Proceedings of the 2nd International Conference on Business Process Management*, volume 3080 of *Lecture Notes in Computer Science*, Potsdam, Germany, June 2004. Springer-Verlag.
30. Z. Duan, A. Bernstein, P. Lewis, and S. Lu. A model for abstract process specification, verification and composition. In M. Aiello, M. Aoyama, F. Curbera, and M.P. Papazoglou, editors, *Proceedings of the 2nd ACM International Conference on Service Oriented Computing*, pages 232–241, New York, NY, USA, November 2004. ACM.
31. Z. Duan, A. Bernstein, P. Lewis, and S. Lu. Semantics based verification and synthesis of BPEL4WS abstract processes. In *Proceedings of the IEEE International Conference on Web Services*, pages 734–737, San Diego, CA, USA, July 2004. IEEE.
32. D. Fahland. Complete abstract operational semantics for the web service business process execution language. Informatik-Berichte 190, Humboldt-Universität zu Berlin, Berlin, Germany, September 2005.
33. D. Fahland and W. Reisig. ASM-based semantics for BPEL: The negative control flow. In D. Beauquier, E. Börger, and A. Slissenko, editors, *Proceedings of the 12th International Workshop on Abstract State Machines*, pages 131–151, Paris, France, March 2005.
34. R. Farahbod. Extending and refining an abstract operational semantics of the web services architecture for the business process execution language. Master's thesis, Simon Fraser University, Burnaby, Canada, July 2004.
35. R. Farahbod, U. Glässer, and M. Vajihollahi. Abstract operational semantics of the business process execution language for web services. Technical Report SFU-CMPT-TR-2004-03, Simon Frasier University, Burnaby, Canada, April 2004.

36. R. Farahbod, U. Glässer, and M. Vajihollahi. Specification and validation of the business process execution language for web services. In W. Zimmermann and B. Thalheim, editors, *Proceedings of the 11th International Workshop on Abstract State Machines*, volume 3052 of *Lecture Notes in Computer Science*, pages 78–94, Lutherstadt Wittenberg, Germany, May 2004. Springer-Verlag.
37. R. Farahbod, U. Glässer, and M. Vajihollahi. An abstract machine architecture for web service based business process management. In C. Bussler and A. Haller, editors, *Proceedings of the Business Process Management Workshops*, volume 3812 of *Lecture Notes in Computer Science*, pages 144–157, Nancy, France, September 2005. Springer-Verlag.
38. R. Farahbod, U. Glässer, and M. Vajihollahi. A formal semantics for the business process execution language for web services. In S. Bevinakoppa, L.F. Pires, and S. Hammoudi, editors, *Proceedings of the Joint Workshop on Web Services and Model-Driven Enterprise Information Services*, pages 122–133, Miami, FL, USA, May 2005. INSTICC Press.
39. J.-C. Fernandez, H. Garavel, A. Kerbrat, L. Mounier, R. Mateescu, and M. Sighireanu. CADP – a protocol validation and verification toolbox. In R. Alur and T. Henzinger, editors, *Proceedings of the 8th International Conference on Computer Aided Verification*, volume 1102 of *Lecture Notes in Computer Science*, pages 437–440, New Brunswick, NJ, USA, July 1996. Springer-Verlag.
40. A. Ferrara. Web services: a process algebra approach. In M. Aiello, M. Aoyama, F. Curbera, and M.P. Papazoglou, editors, *Proceedings of 2nd ACM International Conference on Service Oriented Computing*, pages 242–251, New York, NY, USA, November 2004. ACM.
41. H. Foster. Web service compositions: From XML syntax to service models. In *Proceedings of XML 2005 Conference*, Atlanta, GA, USA, November 2005. RenderX.
42. H. Foster. *A Rigorous Approach To Engineering Web Service Compositions*. PhD thesis, Imperial College, London, UK, January 2006.
43. H. Foster, S. Uchitel, J. Kramer, and J. Magee. Compatibility verification for web service choreography. In *Proceedings of the IEEE International Conference on Web Services*, pages 738–741, San Diego, CA, USA, June 2004. IEEE.
44. H. Foster, S. Uchitel, J. Magee, and J. Kramer. Model-based verification of web service compositions. In *Proceedings of 18th IEEE International Conference on Automated Software Engineering*, pages 152–163, Montreal, Canada, October 2003. IEEE.
45. H. Foster, S. Uchitel, J. Magee, and J. Kramer. Leveraging Eclipse for integrated model-based engineering of web service compositions. In *Proceedings of the 2005 OOPSLA Workshop on Eclipse Technology Exchange*, pages 95–99, San Diego, CA, USA, October 2005. ACM.
46. H. Foster, S. Uchitel, J. Magee, and J. Kramer. Tool support for model-based engineering of web service compositions. In *Proceedings of the 2005 IEEE International Conference on Web Services*, pages 95–102, Orlando, FL, USA, July 2005. IEEE.
47. H. Foster, S. Uchitel, J. Magee, and J. Kramer. LTSA-WS: A tool for model-based verification of web service compositions and choreography. In *Proceeding of the 28th International Conference on Software Engineering*, pages 771–774, Shanghai, China, May 2006. ACM.
48. H. Foster, S. Uchitel, J. Magee, and J. Kramer. Model-based analysis of obligations in web service choreography. In *Proceedings of the International Confer-*

ence on Internet and Web Applications and Services, pages 149–149, Guadeloupe, February 2006. IEEE.

49. H. Foster, S. Uchitel, J. Magee, J. Kramer, and M. Hu. Using a rigorous approach for engineering web service compositions: a case study. In *Proceedings of the 2005 IEEE International Conference on Services Computing*, volume 1, pages 217–224, Orlando, FL, USA, July 2005. IEEE.
50. X. Fu. *Formal Specification and Verification of Asynchronously Communicating Web Services.* PhD thesis, University of California, Santa Barbara, CA, USA, June 2004.
51. X. Fu, T. Bultan, and J. Su. Analysis of interacting BPEL web services. In S.I. Feldman, M. Uretsky, M. Najork, and C.E. Wills, editors, *Proceedings of the 13th International World Wide Web Conference*, pages 621–630, New York, NY, USA, May 2004. ACM.
52. X. Fu, T. Bultan, and J. Su. Model checking XML manipulating software. In G.S. Avrunin and G. Rothermel, editors, *Proceedings of the ACM/SIGSOFT International Symposium on Software Testing and Analysis*, pages 252–262, Boston, MA, USA, July 2004. ACM.
53. X. Fu, T. Bultan, and J. Su. WSAT: A tool for formal analysis of web services. In R. Alur and D. Peled, editors, *Proceedings of the 16th International Conference on Computer Aided Verification*, volume 3114 of *Lecture Notes in Computer Science*, pages 510–514, Boston, MA, USA, July 2004. Springer-Verlag.
54. X. Fu, T. Bultan, and J. Su. Synchronizability of conversations among web services. *IEEE Transactions on Software Engineering*, 31(12):1042–1055, December 2005.
55. A. Fuxman, L. Liu, J. Mylopoulos, M. Pistore, M. Roveri, and P. Traverso. Specifying and analyzing early requirements in Tropos. *Requirements Engineering*, 9(2):132–150, May 2004.
56. S. Haddad, T. Melliti, P. Moreaux, and S. Rampacek. A dense time semantics for web services specifications languages. In *Proceedings of the 1st International Conference on Information and Communication Technologies: from Theory to Applications*, pages 647–648, Damas, Syrie, April 2004. IEEE.
57. S. Haddad, T. Melliti, P. Moreaux, and S. Rampacek. Modelling web services interoperability. In *Proceedings of the 6th International Conference on Enterprise Information Systems*, volume 4, pages 287–295, Porto, Portugal, April 2004.
58. R. Hamadi. *Formal Composition and Recovery Policies in Service-Based Business Processes*. PhD thesis, The University of New South Wales, Sydney, Australia, April 2005.
59. R. Hamadi and B. Benatallah. A Petri net-based model for web service composition. In K. Schewe and X. Zhou, editors, *Proceedings of the 14th Australasian Database Conference*, volume 17 of *CRPITS*, pages 191–200, Adelaide, Australia, February 2003. Australian Computer Society.
60. S. Hinz. Implementierung einer Petrinetz-semantik für BPEL. Master's thesis, Humboldt-Universität zu Berlin, Berlin, Germany, March 2005.
61. S. Hinz, K. Schmidt, and C. Stahl. Transforming BPEL to Petri nets. In W.M.P. van der Aalst, B. Benatallah, F. Casati, and F. Curbera, editors, *Proceedings of the 3rd International Conference on Business Process Management*, volume 2649 of *Lecture Notes in Computer Science*, pages 220–235, Nancy, France, September 2005. Springer-Verlag.
62. G.J. Holzmann. *The Spin Model Checker: Primer and Reference Manual.* Addison-Wesley, Boston, MA, USA, 2004.

63. R. Hull and J. Su. Tools for design of composite web services. In G. Weikum, A.C. König, and S. Deßloch, editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 958–961, Paris, France, June 2004. ACM.
64. R. Hull and J. Su. Tools for composite web services: A short overview. *SIGMOD Record*, 34(2):86–95, June 2005.
65. K. Huynh. Analysis through reflection: walking the EMF model of BPEL4WS. Master's thesis, York University, Toronto, Canada, September 2005.
66. M.B. Juric, B. Mathew, and P. Sarang. *Business Process Execution Language for Web Services: BPEL and BPEL4WS*. Packt, Birmingham, UK, second edition, 2006.
67. R. Kazhamiakin and M. Pistore. Parametric communication model for the verification of BPEL4WS compositions. In M. Bravetti, L. Kloul, and G. Zavattaro, editors, *Proceedings of the 2nd International Workshop on Web Services and Formal Methods*, volume 3670 of *Lecture Notes in Computer Science*, pages 318–332, Versailles, France, September 2005. Springer-Verlag.
68. R. Kazhamiakin, M. Pistore, and M. Roveri. Formal verification of requirements using SPIN: A case study on web services. In *Proceedings of the 2nd International Conference on Software Engineering and Formal Methods*, pages 406–415, Beijing, China, September 2004. IEEE.
69. J. Koehler, G. Tirenni, and S. Kumaran. From business process model to consistent implementation: a case study for formal verification methods. In *Proceedings of the 6th International Enterprise Distributed Object Computing Conference*, pages 96–106, Lausanne, Switzerland, September 2002. IEEE.
70. M. Koshkina. Verification of business processes for web services. Master's thesis, York University, Toronto, Canada, October 2003.
71. M. Koshkina and F. van Breugel. Modelling and verifying web service orchestration by means of the concurrency workbench. *ACM SIGSOFT Software Engineering Notes*, 29(5), September 2004.
72. F. Leymann. Web services flow language (WSFL 1.0), May 2001.
73. N. Lohmann, P. Massuthe, C. Stahl, and D. Weinberg. Analyzing interacting BPEL processes. In *Proceedings of the 4th International Conference on Business Process Management*, Lecture Notes in Computer Science, Vienna, Austria, September 2006. Springer-Verlag.
74. J. Magee and J. Kramer. *Concurrency: State Models and Java Programs*. Wiley, New York, NY, USA, second edition, 2006.
75. A. Martens. *Distributed Business Processes — Modeling and Verification by help of Web Services*. PhD thesis, Humboldt-Universität zu Berlin, Berlin, Germany, July 2003.
76. A. Martens. On compatibility of web services. *Petri Net Newsletter*, 65:12–20, October 2003.
77. A. Martens. On usability of web services. In C. Calero, O. Diaz, and M. Piattini, editors, *Proceedings of 4th International Conference on Web Information Systems Engineering Workshops*, pages 182–190, Rome, Italy, December 2003. IEEE.
78. A. Martens. Analysis and re-engineering of web services. In *Proceedings of the 6th International Conference on Enterprise Information Systems*, pages 419–426, Porto, Portugal, April 2004.
79. A. Martens. *Verteilte Geschäftsprozesse – Modellierung und Verifikation mit Hilfe von Web Services*. WiKu-Verlag, 2004.

80. A. Martens. Analyzing web service based business processes. In M. Cerioli, editor, *Proceedings of the 8th International Conference on Fundamental Approaches to Software Engineering*, volume 3442 of *Lecture Notes in Computer Science*, pages 19–33, Edinburgh, UK, April 2005. Springer-Verlag.
81. A. Martens. Consistency between executable and abstract processes. In *Proceedings of the 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service*, pages 60–67, Hong Kong, March/April 2005. IEEE.
82. A. Martens. Simulation and equivalence between BPEL process models. In D. Tutsch, editor, *Proceedings of the Design, Analysis, and Simulation of Distributed Systems Symposium*, San Diego, CA, USA, April 2005. The Society for Modeling and Simulation International.
83. A. Martens and S. Moser. Diagnosing SCA components using WOMBAT. In *Proceedings of the 4th International Conference on Business Process Management*, Lecture Notes in Computer Science, Vienna, Austria, September 2006. Springer-Verlag.
84. A. Martens, S. Moser, A. Gerhardt, and K. Funk. Analyzing compatibility of BPEL processes. In *Proceedings of the Advanced International Conference on Telecommunications and the International Conference on Internet and Web Applications and Services*, page 147, Guadeloupe, February 2006. IEEE.
85. M. Mazarra and R. Lucchi. A framework for generic error handling in business processes. In *Proceedings of the 1st International Workshop on Web Services and Formal Method*, volume 105 of *Electronic Notes in Theoretical Computer Science*, pages 133–145, Pisa, Italy, February 2004. Elsevier.
86. K.L. McMillan. *Symbolic Model Checking: An Approach to the State Explosion Problem*. Kluwer, Boston, MA, USA, 1993.
87. R. Milner. *Communication and Concurrency*. Prentice Hall, New York, NY, USA, 1989.
88. R. Milner. *Communicating and Mobile Systems: The π-Calculus*. Cambridge University Press, Cambridge, UK, 1999.
89. S. Nakajima. On verifying web service flows. In *Proceedings of the Symposium on Applications and the Internet Workshops*, pages 223–224, Nara City, Japan, January/February 2002. IEEE.
90. S. Nakajima. Verification of web service flows with model-checking techniques. In *Proceedings of the 1st International Symposium on Cyber Worlds*, pages 378–386, Tokyo, Japan, November 2002. IEEE.
91. S. Nakajima. Model-checking of safety and security aspects in web service flows. In N. Koch, P. Fraternali, and M. Wirsing, editors, *Proceedings of the 4th International Conference on Web Engineering*, volume 3140 of *Lecture Notes in Computer Science*, pages 488–501, Munich, Germany, July 2004. Springer-Verlag.
92. S. Nakajima. Lightweight formal analysis of web service flows. *Progress in Informatics*, 1(2):57–76, November 2005.
93. S. Nakajima. Model-checking behavioral specification of BPEL applications. In *Proceedings of the International Workshop on Web Languages and Formal Methods*, volume 151(2) of *Electronic Notes in Theoretical Computer Science*, pages 89–105, New Castle, UK, July 2005. Elsevier.
94. C. Ouyang, W.M.P. van der Aalst, S. Breutel, M. Dumas, A.H.M. ter Hofstede, and E. Verbeek. WofBPEL: A tool for automated analysis of BPEL processes. In B. Benatallah, F. Casati, and P. Traverso, editors, *Proceedings of the 3rd International Conference on Service-Oriented Computing*, volume 3826 of *Lecture Notes in Computer Science*, pages 484–489, Amsterdam, The Netherlands, December 2005. Springer-Verlag.

95. C. Ouyang, W.M.P. van der Aalst, S. Breutel, M. Dumas, A.H.M. ter Hofstede, and H.M.W. Verbeek. Formal semantics and analysis of control flow in WS-BPEL. Report BPM-05-15, BPM Center, 2005.
96. F.G. Pagan. *Formal Specification of Programming Languages*. Prentice Hall, New York, NY, USA, 1981.
97. M. Pistore, M. Roveri, and P. Busetta. Requirements-driven verification of web services. In *Proceedings of the 1st International Workshop on Web Services and Formal Method*, volume 105 of *Electronic Notes in Theoretical Computer Science*, pages 95–108, Pisa, Italy, February 2004. Elsevier.
98. M. Pistore, P. Traverso, P. Bertoli, and A. Marconi. Automated synthesis of composite BPEL4WS web services. In *Proceedings of the 2005 IEEE International Conference on Web Services*, pages 293–301, Orlando, FL, USA, July 2005. IEEE.
99. G. Pu, X. Zhao, S. Wang, and Z. Qiu. Towards the semantics and verification of BPEL4WS. In *Proceedings of the International Workshop on Web Languages and Formal Methods*, volume 151(2) of *Electronic Notes in Theoretical Computer Science*, pages 33–52, New Castle, UK, July 2005. Elsevier.
100. G. Pu, H. Zhu, Z. Qiu, S. Wang, X. Zhao, and J. He. Theoretical foundation of scope-based compensable flow language for web service. In R. Gorrieri and H. Wehrheim, editors, *Proceedings of the 8th IFIP WG 6.1 International Conference on Formal Methods for Open Object-Based Distributed Systems*, volume 4037 of *Lecture Notes in Computer Science*, pages 251–266, Bologna, Italy, June 2006. Springer-Verlag.
101. Z. Qiu, S. Wang, G. Pu, and X. Zhao. Semantics of BPEL4WS-like fault and compensation handling. In J. Fitzgerald, I.J. Hayes, and A. Tarlecki, editors, *Proceedings of the International Symposium on Formal Methods*, volume 3582 of *Lecture Notes in Computer Science*, pages 350–365, Newcastle upon Tyne, UK, July 2005. Springer-Verlag.
102. W. Reisig. *Petri Nets: An Introduction*. Springer-Verlag, New York, NY, USA, 1985.
103. W. Reisig. Modeling- and analysis techniques for web services and business processes. In M. Steffen and G. Zavattaro, editors, *Proceedings of the 7th IFIP WG 6.1 International Conference on Formal Methods for Open Object-Based Distributed Systems*, volume 3535 of *Lecture Notes in Computer Science*, pages 243–258, Athens, Greece, June 2005. Springer-Verlag.
104. W. Reisig, K. Schmidt, and C. Stahl. Kommunizierende workflow-services modellieren und analysieren. *Informatik - Forschung und Entwicklung*, 20(1/2):90–101, October 2005.
105. G. Salaün, L. Bordeaux, and M. Schaerf. Describing and reasoning on web services using process algebra. In *Proceedings of the IEEE International Conference on Web Services*, pages 43–51, San Diego, CA, USA, June 2004. IEEE.
106. G. Salaün, A. Ferrara, and A. Chirichiello. Negotiation among web services using LOTOS/CADP. In L. Zhang, editor, *Proceedings of the European Conference on Web Services*, volume 3250 of *Lecture Notes in Computer Science*, pages 198–212, Erfurt, Germany, September 2004. Springer-Verlag.
107. B. Schlingloff, A. Martens, and K. Schmidt. Modeling and model checking web services. In *Proceedings of the 2nd International Workshop on Logic and Communication in Multi-Agent Systems*, volume 126 of *Electronic Notes in Theoretical Computer Science*, pages 3–26, Nancy, France, August 2004. Elsevier.
108. K. Schmidt. LoLA – a low level analyser. In M. Nielsen and D. Simpson, editors, *Proceedings of the 21st International Conference on Application and Theory of*

Petri Nets, volume 1825 of *Lecture Notes in Computer Science*, pages 465–474, Aarhus, Denmark, June 2000. Springer-Verlag.

109. K. Schmidt and C. Stahl. A Petri net semantic for BPEL4WS – validation and application. In E. Kindler, editor, *Proceedings of the 11th Workshop on Algorithms and Tools for Petri Nets*, pages 1–6, Paderborn, Germany, September/October 2004. Universität Paderborn.
110. C. Stahl. Transformation von BPEL4WS in Petrinetze. Master's thesis, Humboldt-Universität zu Berlin, Berlin, Germany, April 2004.
111. S. Thatte. XLANG, 2001.
112. G. Tremblay and J. Chae. Towards specifying contracts and protocols for web services. In H. Mili and F. Khendek, editors, *Proceedings of the MCeTech Montreal Conference on eTechnologies*, pages 73–85, Montreal, Canada, January 2005.
113. H.M.W. Verbeek and W.M.P. van der Aalst. Analyzing BPEL processes using Petri nets. In D. Marinescu, editor, *Proceedings of the Second International Workshop on Applications of Petri Nets to Coordination, Workflow and Business Process Management*, pages 59–78, Miami, FL, USA, October 2005.
114. H.M.W. Verbeek, T. Basten, and W.M.P. van der Aalst. Diagnosing workflow processes using Woflan. *The Computer Journal*, 44(4):246–279, July 2001.
115. M. Viroli. Towards a formal foundation to orchestration languages. In *Proceedings of the 1st International Workshop on Web Services and Formal Method*, volume 105 of *Electronic Notes in Theoretical Computer Science*, pages 51–71, Pisa, Italy, February 2004. Elsevier.
116. A. Wombacher, P. Fankhauser, and E. Neuhold. Transforming BPEL into annotated deterministic finite state automata for service discovery. In *Proceedings of the 2nd IEEE International Conference on Web Services*, pages 316–323, San Diego, CA, USA, July 2004. IEEE.
117. Y. Yang, Q. Tan, and Y. Xiao. Verifying web services composition. In J. Akoka, S.W. Liddle, I. Song, M. Bertolotto, I. Comyn-Wattiau, S.S. Cherfi, W. vanden Heuvel, B. Thalheim, M. Kolp, P. Bresciani, J. Trujillo, C. Kop, and H.C. Mayr, editors, *Proceedings of the ER 2005 Workshops*, volume 3770 of *Lecture Notes in Computer Science*, pages 354–363, Klagenfurt, Austria, October 2005. Springer-Verlag.
118. Y. Yang, Q. Tan, and Y. Xiao. Verifying web services composition based on hierarchical colored Petri nets. In *Proceedings of the 1st International Workshop on Interoperability of Heterogeneous Information Systems*, pages 47–54, Bremen, Germany, November 2005. ACM.
119. Y. Yang, Q. Tan, Y. Xiao, J. Yu, and F. Liu. Exploiting hierarchical CP-nets to increase reliability of web services workflow. In *Proceeding of the International Symposium on Applications and the Internet*, pages 116–122, Phoeniz, AZ, USA, January 2006. IEEE.
120. Y. Yang, Q. Tan, Xiao Y, J. Yu, and F. Liu. Verifying web services composition: A transformation-based approach. In *Proceedings of the 6th International Conference on Parallel and Distributed Computing*, pages 546–548, Dalian, China, December 2005. IEEE.
121. Y. Yang, Q. Tan, J. Yu, and F. Liu. Transformation BPEL to CP-nets for verifying web services composition. In *Proceedings of the International Conference on Next Generation Web Services Practices*, Seoul, Korea, August 2005. IEEE.
122. X. Yi and K.J. Kochut. A CP-nets-based design and verification framework for web services composition. In *Proceedings of the 2nd IEEE International Conference on Web Services*, pages 756–760, San Diego, CA, USA, July 2004. IEEE.

123. X. Yi and K.J. Kochut. Process composition of web services with complex conversation protocols: a colored Petri nets based approach. In *Proceedings of the Design, Analysis, and Simulation of Distributed Systems Symposium*, pages 141–148, Arlington, VA, USA, April 2004. The Society for Modeling and Simulation International.
124. L.-J. Zhang, editor. *Proceedings of the International Conference on Web Services*, Las Vegas, NV, USA, June 2003. CSREA Press.