A Survey of Formal Verification for Business Process Modeling

Shoichi Morimoto

School of Industrial Technology, Advanced Institute of Industrial Technology
1-10-40, Higashi-oi, Shinagawa-ku, Tokyo, 140-0011, Japan
morimoto-syoichi@aiit.ac.jp

Abstract. Information systems have to respond well to the changing business environment. Thus, they must have architecture which withstands the change. To design such systems, business process modeling is effective, however, the models include often abstractness and arbitrariness. Therefore, there have been efforts that validate rigorousness of the models. They have defined semantics of the models and applied various logics and formal methods to verification of the rigorousness. This paper focuses on formal verification of the models and surveys the efforts. We also discuss the prospect of the solutions. The establishment of the verification will be surely helpful toward solving the problems on business process reengineering, business process management, service-oriented architecture, and so on.

1 Introduction

Recently, enterprise information systems are designed based on service-oriented architecture. The solution against the changing business environment is construction of flexible business processes, which is the core of enterprise information systems development. It is common knowledge that business process modeling (BPM) is effective for the development. Developers can generally model business processes with modeling notation, e.g., BPMN [38], activity diagrams of UML [22]. The diagram, modeled with the notation, is simple and intuitively understandable at a glance. The notation is also designed so that anyone can easily model. Moreover, the notation is closely relevant to web services; the diagram can be converted into the BPEL XML format [40].

However, work of general modeling includes arbitrariness and lacks strictness. A diagram modeled with the notation may have various interpretations and one or more different diagrams may denote one process. Thus, before utilizing BPM, we must define strict semantics of the models and verify formally them. There have been many efforts that validate strictness of the diagrams; automation tools which can debug grammatical errors of BPMN and convert diagrams into BPEL [23], formal methods for verifying diagrams based on the π calculus [34] or Petri Net [42], techniques proving consistency with model-checking [5], and so on.

In this paper we present a survey of existing proposals for formal verification techniques of business process diagrams and compare them among each other with respect to motivations, methods, and logics. We also discuss some conclusive considerations and our direction for future work. We hope the survey contributes to designers and developers of enterprise information systems for solving issues on their section and satisfying the industrial needs.

2 Formal Verification of Business Process Models

2.1 Basic Logics of the Verification

To verify formally business process models, it is firstly required to give the formal semantics to the models. The logical bases and researches using them are roughly classified as follows.

Automata. Automata are a public and base model of formal specifications for systems [21]. An automaton consists of a set of states, actions, transitions between states, and an initial state. Labels denote the transition from one state to another. Many specification models to express system behavior derive from automata.

In the reference [16], the authors propose a framework to analyze and verify properties of BPMN diagrams converted into the BPEL format that communicate via asynchronous XML messages. The framework first converts the processes to a particular type of automata whose every transition is equipped with a guard in an XPath format, after which these guarded automata are translated into Promela (Process or Protocol Meta Language) for the SPIN model checker [20]. Consequently, SPIN can be used to verify whether business process models satisfy properties formalized in LTL (Linear Time Temporal Logic).

In the reference [8], the authors show a case study to convert automatically business processes written in BPEL-WSCDL to timed automata and to verify subsequently them by the UPPAAL model checker [49]. The authors are currently implementing a tool for the automatic translation that utilizes UPPAAL.

In the reference [10], the authors propose a framework to verify automatically business processes that are modelled in Orc [35]. The authors define a formal timed-automata semantics for Orc expressions, which verifies to the Orc's operational semantics. Accordingly, one can verify formally Orc models with UPPAAL. The paper also shows a simple case study.

Thus, to verify business process diagrams the efforts utilizing automata convert the diagrams to XML formats (e.g., BPEL, XPDL, WS-CDL, Orc) for the present. After that, they accommodate automata to XML formats and then model-checking tools can verify them (Fig. 1). Besides the above automata models, team automata [12] and I/O automata [30] may be helpful for the verification. Team automata allow one to specify separately the components of a system, to describe their interactions, and to reuse the components. Their advantage is a flexible description for communication services among distributed systems, extending I/O automata. This advantage enables team automata to describe the formal model of secure web service compositions.

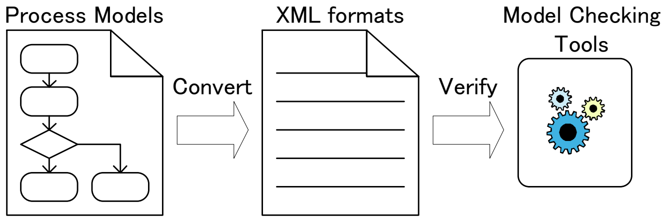

Fig. 1. Verification of business process models with automata

Petri Net. Petri Net is a framework to model concurrent systems. Petri Net can identify many basic aspects of concurrent systems simply, mathematically and conceptually. Therefore, many theories of concurrent systems derive from Petri Net. Moreover, because Petri Net has easily understandable and graphical notation, it has been widely applied.

Petri Net often become a topic in BPM and is related to capturing process control flows [50]. Petri Net can specially detect the dead path of business process models whose preconditions are not satisfied. The paper [9] shows how to correspond all BPMN diagrams constructs into labeled Petri Net. This output can subsequently be used to verify BPEL processes by the open source tools BPEL2PNML and WofBPEL [41].

In the reference [36], the authors define the semantics of relation BPEL and OWL-S [51] in terms of first-order logic. Based on this semantics they formalize business processes in Petri Net, complete with an operational semantics. They also develop a tool to describe and automatically verify composition of business processes.

In the reference [17], the authors apply a Petri-net-based algebra to modeling business processes, based on control flows.

The paper [52] proposes a Petri-net-based design and verification tool for web service composition. The tool can visualize, create, and verify business processes. The authors are now improving the graphical user interface which can be used to aid the business process modeling and to edit Petri Net and BPEL in a lump.

The paper [53] introduce a Petri-net-based architectural description language, named WS-Net, in which web-service-oriented systems can be modeled, and presents a simple example. To handle real applications and to detect errors in business processes, the authors are currently developing an automatic translation tool from WSDL to WS-Net.

The paper [18] proposes a formal Petri Net semantics for BPEL which assures exception handling and compensations. Moreover, the authors present the parser which can automatically convert business process diagrams into Petri Net. Consequently, the semantics enabled many Petri Net verification tools to automatically analyze business processes.

In the reference[45], the authors propose a framework which can translate Orc into colored Petri Net. Colored Petri Net has been proposed to model large scale systems more effectively. The framework and tool deal with recursion and data handling. Moreover, because the framework and tool can simulate and verify the behavior of process models at the design phase of information systems, users of them can detect and correct errors beforehand. Therefore, they contribute to raise the reliability of business process diagrams.

Petri Net is the traditional and well-established technique, thus there have been many verification methods and tools. The essentials of the above efforts are how to translate business process diagrams into Petri Net. After that, we have a rich variety of tools for the verification. However, all the components in business process modeling notation cannot change into Petri Net. For instance, BPMN has various gateways, event triggers, loop activities, control flows, and nested/embedded sub-processes. It is difficult to define the correspondence of these objects to Petri Net. There is room for argument on the translation.

Process Algebras. Process algebras are a diverse family of related approaches to formally modeling concurrent systems. Their semantic foundation is based on automata. Many derivative algebras have been defined and many references about them exists. The representative process algebras are Milner's Calculus of Communicating Systems (CCS [33]), Hoare's Calculus of Sequential Processes (CSP [19]), the Algebra of Communicating Processes (ACP [1]) by Bergstra and Klop, and the Language of Temporal Ordered Systems (LOTOS [2]) ISO standard. Process algebras are strict and well-established theories that support the automatic verification of properties of systems behavior as well as Petri Net. They also provide a rich theory on bisimulation analysis. The analysis is helpful to verify whether a service can substitute another service in a composition or the redundancy of a service[3]. Among these, the π calculus is a process algebra that influences business process description languages, e.g., XLANG, BPEL. In respect of automatic verification, the π calculus is far superior to Petri Net. From a compositional perspective, the π calculus offers constructs to compose activities in terms of sequential, parallel, and conditional execution, combinations of which can lead to compositions of arbitrary complexity. The followings are process-algebraic approaches to specify and verify secure compositions of business processes.

In the reference [46], the authors discuss the application of process algebras to describe, compose, and verify business processes, with a particular focus on their interactions. They show an example in which they use CCS to specify and compose business processes. They also use the Concurrency Workbench [6] to validate properties such as correct business process composition. If the π calculus is used instead of CCS, this approach can be useful in practical application. It may solve real issues, e.g., the exchange of messages during business process interactions.

In the reference [14], the authors define correspondence between BPEL and LOTOS. The advantage of this proposal is that it includes compensations and exception handling. Thus, it enables the verification of temporal properties with the CADP model checker [13].

Thus, process algebraic approaches are suitable for verification of the reliability of information systems, because they can simulate the behavior of business process models and correct their error at the design phase as well as Petri-net-based verification.

2.2 Aims of the Verification

We also classified the researches from the perspective of the aim of the verification. The followings are properties which are elicited from the above researches.

Connectivity. We must guarantee connectivity by the verification. Developers can determine which processes are composed and reason about their interactions with the reliable connectivity. On the other hand, they must also satisfy non-functional requirements, e.g., timeliness, security, and dependability. It is important in a B2B system to define consensus between involved stakeholders. Business consensus defines the contract between two or more stakeholders on such requirements. It is necessary to describe and verify the requirements in composed business processes.

Correctness. With the scale-spreading and diversification of information systems, concurrency of large systems are increasingly important. The correctness of such large systems corresponds to their temporal behavior. Behavioral properties can be classified into safety and liveness. A safety property stipulates that "nothing bad" will happen, ever, during the execution of a system, and a liveness property stipulates that "something good" will happen, eventually, during the execution of a system. Formal verification provides rigorous and mathematical semantics of guaranteeing large systems to comply with their specification.

Compensation and Scalability. In the real-world business, end users maybe generally want to interact with many services, thus enterprise information systems must be connected with possibly many services. Therefore, one of the important verification properties is how many services can business process models implement.

Compatibility. Business process modeling notation intends to bridge the gap between business process design and implementation of enterprise information systems. The diagrams should be verified process interactions for business collaboration described in such notation. Business collaboration must satisfy compatibility between business participants in the collaboration.

3 Discussion

With the above verification techniques, verifiers must formalize mathematically business process models and verification criteria, considering the semantics and various issues of their business. This is difficult for verifiers who are not specialists of formal methods or conversant with the methods. They may make mistakes in the formal descriptions of verification criteria, or may not know how to formally describe the verification criteria. Moreover, they must define the verification criteria themselves, but such criteria does not have objectivity. Thus, it is desirable to establish objective criteria for business process models, e.g., ISO standard.

The essential purpose of BPM is to construct processes which yield a profit for enterprise. Thus, we should verify not only the above properties of the diagrams but also the profit which is generated by the model. That is, we must verify whether the process model certainly yields a profit. We are now discussing whether the chance discovery process [39] can be applied to the verification from the financial and business administration viewpoint. Moreover, various logics introducing probability into first-order predicate logic have been proposed [7][11][15][24][26][27][32][37][43][44]. These studies make it possible to generate Bayesian Network [25] from predicate logic expression based on knowledge-based model construction [4]. Since there are some business processes which flow with non-programmable decision [48], the business process diagrams with uncertainness have to be verified by such logics. If the correspondence of the business process modeling notations to Bayesian Network is defined (Fig. 2), we can verify the diagrams based on probabilistic inference.

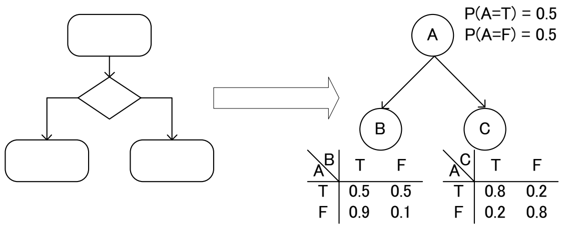

Fig. 2. An example of the correspondence business process models to Bayesian networks

4 Conclusion

In this paper, we have presented the formal verification techniques which simulate and verify one's business process models at the design phase of enterprise information systems. These techniques can detect and correct errors of the models as early as possible and in any case before implementation. We have also shown future work of the formal verification for BPM. However, the comparison in this paper surveyed only the basic logics and the aims of the verification, thus we must also define the other quantitative information in order to choose which logics or methods better suit the formal verification of BPM. Moreover, there are the well-established practices which verify UML state machine diagrams for behavior with model-checking [28][29][31][47]. We should compare BPM verification with such studies.

A prospect of the researches that we would like to deepen in future work is to determine the financial characteristics that each of the languages and models is able to describe in order to define a valuable business process.

References

1. Bergstra, J.A., Klop, J.W.: Algebra of Communicating Processes with Abstraction. In: Theor. Comput. Sci., vol. 37, pp. 77–121. Elsevier, Amsterdam (1985)
2. Bolognesi, T., Brinksma, E.: Introduction to the ISO Specification Language LOTOS. Computer Networks 14, 25–59 (1987)
3. Bordeaux, L., Salaün, G., Berardi, D., Mecella, M.: When are Two Web Services Compatible? In: Shan, M.-C., Dayal, U., Hsu, M. (eds.) TES 2004. LNCS, vol. 3324, pp. 15–28. Springer, Heidelberg (2005)
4. Breese, J.S.: Construction of Belief and Decision Networks. Computational Intelligence 8(4), 624–647 (1992)
5. Clarke, E., Grumberg, O., Peled, D.: Model Checking. MIT Press, Cambridge (2000)
6. Cleaveland, R., Li, T., Sims, S.: The Concurrency Workbench of the New Century (2000), `http://www.cs.sunysb.edu/~cwb/`
7. Cussens, J.: Parameter Estimation in Stochastic Logic Programs. Machine Learning 44(3), 245–271 (2001)
8. Díaz, G., Pardo, J.J., Cambronero, M.-E., Valero, V., Cuartero, F.: Automatic Translation of WS-CDL Choreographies to Timed Automata. In: Bravetti, M., Kloul, L., Zavattaro, G. (eds.) EPEW/WS-EM 2005. LNCS, vol. 3670, pp. 230–242. Springer, Heidelberg (2005)
9. Dijkman, R.M., Dumas, M., Ouyang, C.: Formal Semantics and Analysis of BPMN Process Models using Petri Nets, Queensland University of Technology (2007), `http://eprints.qut.edu.au/archive/00007115/`
10. Dong, J.S., Liu, Y., Sun, J., Zhang, X.: Verification of Computation Orchestration via Timed Automata. In: Liu, Z., He, J. (eds.) ICFEM 2006. LNCS, vol. 4260, pp. 226–245. Springer, Heidelberg (2006)
11. Eisner, J., Goldlust, E., Smith, N.A.: Compiling Comp Ling: Weighted Dynamic Programming and the Dyna Language, Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing. In: Proc. of the Conference (HLT/EMNLP 2005). The Association for Computational Linguistics, pp. 281–290 (2005)
12. Ellis, C.: Team Automata for Groupware Systems. In: Proc. of the international ACM SIGGROUP conference on Supporting group work (GROUP 1997), pp. 415–424. ACM Press, New York (1997)
13. Fernandez, J.-C., Garavel, H., Kerbrat, A., Mounier, L., Mateescu, R., Sighireanu, M.: CADP - A Protocol Validation and Verification Toolbox. In: Alur, R., Henzinger, T.A. (eds.) CAV 1996. LNCS, vol. 1102, pp. 437–440. Springer, Heidelberg (1996)
14. Ferrara, A.: Web Services: A Process Algebra Approach. In: Proc. of the 2nd International Conference on Service-Oriented Computing (ICSOC 2004), pp. 242–251. ACM Press, New York (2004)

15. Friedman, N., Getoor, L., Koller, D., Pfeffer, A.: Learning Probabilistic Relational Models. In: Proc. of the 16th International Joint Conference on Artificial Intelligence (IJCAI 1999), pp. 1300–1309. Morgan Kaufmann, San Francisco (1999)
16. Fu, X., Bultan, T., Su, J.: Analysis of Interacting BPEL Web Services. In: Proc. of the 13th International Conference on the World Wide Web (WWW 2004), pp. 621–630. ACM Press, New York (2004)
17. Hamadi, R., Benatallah, B.: A Petri Net-based Model for Web Service Composition. In: Proc. of the 14th Australasian Database Conference (ADC 2003), Conferences in Research and Practice in Information Technology, vol. 17, pp. 191–200. Australian Computer Society (2003)
18. Hinz, S., Schmidt, K., Stahl, C.: Transforming BPEL to Petri Nets. In: van der Aalst, W.M.P., Benatallah, B., Casati, F., Curbera, F. (eds.) BPM 2005. LNCS, vol. 3649, pp. 220–235. Springer, Heidelberg (2005)
19. Hoare., C.: Communicating Sequential Processes. Prentice-Hall, Englewood Cliffs (1985)
20. Holzmann, G.J.: The SPIN Model Checker –Primer and Reference Manual. Addison-Wesley, Reading (2003)
21. Hopcroft, J.E., Motwani, R., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation, 3rd edn. Addison-Wesley, Reading (2006)
22. ISO/IEC 19501 Standard: Information Technology – Open Distributed Processing – Unified Modeling Language (UML) Version 1.4.2 (2005)
23. ITP Commerce, Process Modeler for Microsoft Visio^{TM}, `http://www.itp-commerce.com/`
24. Jaeger, M.: Relational Bayesian Networks. In: Proc. of the 13th Conference on Uncertainty in Artificial Intelligence (UAI 1997), pp. 266–273. Morgan Kaufmann, San Francisco (1997)
25. Jensen, F.V.: Bayesian Networks and Decision Graphs. Springer, Heidelberg (2001)
26. Kersting, K., Raedt, L.D.: Bayesian Logic Programs. In: Cussens, J., Frisch, A.M. (eds.) ILP 2000. LNCS (LNAI), vol. 1866, Springer, Heidelberg (2000)
27. Koller, D., Pfeffer, A.: Learning Probabilities for Noisy First-Order Rules. In: Proc. of the 15th International Joint Conference on Artificial Intelligence (IJCAI 1997), pp. 1316–1323. Morgan Kaufmann, San Francisco (1997)
28. Latella, D., Majzik, I., Massink, M.: Automatic Verification of a Behavioural Subset of UML Statechart Diagrams using the SPIN Model Checker. Formal Asp. Comput. 11(6), 637–664 (1999)
29. Lilius, J., Paltor, I.: vUML: A Tool for Verifying UML Models. In: Proc. of the 14th IEEE International Conference on Automated Software Engineering (ASE 1999), pp. 255–258. IEEE Computer Society, Los Alamitos (1999)
30. Lynch, N.A., Tuttle, M.R.: An Introduction to Input/Output Automata. CWI Quarterly 2(3), 219–246 (1989)
31. Mikk, E., Lakhnech, Y., Siegel, M., Holzmann, G.J.: Implementing Statecharts in Promela/SPIN. In: Proc. of the 2nd Workshop on Industrial-Strength Formal Specification Techniques (WIFT 1998), pp. 90–101. IEEE Computer Society, Los Alamitos (1998)
32. Milch, B., Marthi, B., Russell, S.J., Sontag, D., Ong, D.L., Kolobov, A.: BLOG: Probabilistic Models with Unknown Objects. In: Proc. of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005), pp. 1352–1359. Professional Book Center (2005)
33. Milner, R.: Communication and Concurrency. Prentice-Hall, Englewood Cliffs (1989)

34. Milner, R.: Communicating and Mobile Systems: The Pi-Calculus. Cambridge University Press, Cambridge (1999)
35. Misra, J., Cook, W.R.: Orc - An Orchestration Language, `http://www.cs.utexas.edu/~wcook/projects/orc/`
36. Narayanan, S., McIlraith, S.A.: Simulation, Verification and Automated Composition of Web Services. In: Proc. of the 11th International World Wide Web Conference (WWW 2002), pp. 77–88. ACM Press, New York (2002)
37. Ngo, L., Haddawy, P.: Answering Queries from Context-Sensitive Probabilistic Knowledge Bases. Theor. Comput. Sci 171(1-2), 147–177 (1997)
38. Object Management Group: Business Process Modeling Notation Specification, Final Adopted Specification dtc/06-02-01 (2006)
39. Ohsawa, Y.: Chance Discoveries for Making Decisions in Complex Real World. New Generation Computing 20(2), 143–163 (2002)
40. Organization for the Advancement of Structured Information Standards: Web Services Business Process Execution Language Version 2.0 (2007)
41. Ouyang, C., Verbeek, E., van der Aalst, W.M.P., Breutel, S., Dumas, M., ter Hofstede, A.H.M.: WofBPEL: A Tool for Automated Analysis of BPEL Processes. In: Benatallah, B., Casati, F., Traverso, P. (eds.) ICSOC 2005. LNCS, vol. 3826, pp. 484–489. Springer, Heidelberg (2005)
42. Petri, C.A.: Kommunikation mit Automaten. PhD thesis, Rheinisch-Westfälisches Institut für Instrumentelle Mathematik an der Universität Bonn (1962)
43. Poole, D.: The Independent Choice Logic for Modelling Multiple Agents under Uncertainty. Artif. Intell. 94(1-2), 7–56 (1997)
44. Richardson, M., Domingos, P.: Markov Logic Networks. Machine Learning 62(1-2), 107–136 (2006)
45. Rosario, S., Benveniste, A., Haar, S., Jard, C.: Net System Semantics of Web Services Orchestrations Modeled in Orc, Research Report IRISA, No. 1780, Istitut de Rechercheen Informatique et Systèmes Aléatoires (2006)
46. Salaün, G., Bordeaux, L., Schaerf, M.: Describing and Reasoning on Web Services using Process Algebra. In: Proc. of the International Conference on Web Services (ICWS 2004), pp. 43–50. IEEE Computer Society, Los Alamitos (2004)
47. Schäfer, T., Knapp, A.,, Merz, S.: Model Checking UML State Machines and Collaborations. Electr. Notes Theor. Comput. Sci. 55(3), 357–369 (2001)
48. Simon, H.A.: The New Science of Management Decision. Prentice-Hall, Englewood Cliffs (1977)
49. UPPAAL: `http://www.uppaal.com/`
50. Wohed, P., van der Aalst, W.M.P., Dumas, M., ter Hofstede, A.H.M.: Analysis of Web Services Composition Languages: The Case of BPEL4WS. In: Song, I.-Y., Liddle, S.W., Ling, T.-W., Scheuermann, P. (eds.) ER 2003. LNCS, vol. 2813, pp. 200–215. Springer, Heidelberg (2003)
51. World Wide Web Consortium: OWL-S: Semantic Markup for Web Services, `http://www.w3.org/Submission/OWL-S/`
52. Yi, X., Kochut, K.: A CP-nets-based Design and Verification Framework for Web Services Composition. In: Proc. of the International Conference on Web Services (ICWS 2004), pp. 756–760. IEEE Computer Society, Los Alamitos (2004)
53. Zhang, J., Chung, J.-Y., Chang, C.K., Kim, S.: WS-Net: A Petri-net Based Specification Model for Web Services. In: Proc. of the International Conference on Web Services (ICWS 2004), pp. 420–427. IEEE Computer Society, Los Alamitos (2004)