Intelligenza Artificiale 5 (2011) 101–105
DOI 10.3233/IA-2011-0011
IOS Press

Modeling and verifying business processes and choreographies through the abductive proof procedure SCIFF and its extensions

Federico Chesani, Paola Mello*, Marco Montali and Paolo Torroni
DEIS, University of Bologna, Viale Risorgimento 2, Bologna, Italy

Abstract. In this article we survey our recent research activity concerning the use of logic programming, and in particular of abduction, for interaction specification and verification in several domains. We outline relevant results in the areas of multi-agent systems, argumentation, web services choreographies and business processes.

Keywords: Logic programming, hypothetical reasoning, interaction, modelling, verification, multi-agent systems, protocols, business processes, web services, choreographies, semantic web, argumentation

1. Background

More than twenty years ago, a significant part of the Italian and European Computer Science research community and IT industry expressed great interest in the new Logic Programming (LP) paradigm. The pioneering research being carried out by Alberto Martelli and by others, significantly contributed to foster such an interest. The University of Bologna's Computing Department's Artificial Intelligence group was born at the time of the LP wave of the Eighties. Our group was attracted by the unique features of LP, mainly by its ability to marry formal and practical aspects, and to conjugate a declarative language with an underlying execution model.

Our group's main research directions back then were centered around distribution, modularity, parallelism, and language extensions, such as Constraint and Abductive Logic Programming. In order to enable the "programming in the large" model within the LP paradigm, two approaches were studied for structuring logic programs: an algebraic method based on meta-operators, and another approach based on language extensions. The first model brought to the definition of an extended LP language called StructuredProlog [17]. This language integrates blocks, modules, hypothetical reasoning, logical theory and object taxonomies, and is implemented as an extension of the Warren Abstract Machine. The second approach was based on the introduction of negation in LP to support non-monotonic reasoning.

Past research activities also focussed on parallel logic languages with *AND* parallelism and no variable sharing on a *MIMD* architecture. Inter-process communication and synchronization were possible via multi-headed clauses and a shared blackboard, and via an optimized unification mechanism specifically tailored to serve the purpose. Finally, the LP paradigm has been integrated with the OO paradigm, to define the Distributed Logic Objects language (DLO). In DLO, methods are expressed via multi-headed clauses, in a purely

*Corresponding author. DEIS, University of Bologna, Viale Risorgimento 2, 40136 - Bologna, Italy. E-mails: paola.mello@unibo.it (P. Mello); federico.chesani@unibo.it (F. Chesani); marco.montali@unibo.it (M. Montali); paolo.torroni@unibo.it (P. Torroni).

1724-8035/11/$27.50 © 2011 – IOS Press and the authors. All rights reserved

declarative style, while specific constructs are defined to express interaction among objects and inheritance.

2. The SOCS project

Since 2001, our group has invested many resources in the advancement of the state of the art of computational logic-based multi-agent systems [21], specifically targeting agent interaction. The aim was to develop an LP-based language and an operational model for the specification and verification of agent interaction protocols. Such work has been carried out in the context of the EU-funded SOCS project.[1]

The SOCS society model [25, 4], developed by a joint effort between the Universities of Bologna and Ferrara, gives concrete guidelines for the formal specification of interaction among agents that form a society, and for the definition of a computational logic-based architecture for agent interaction. In the proposed architecture, the society defines the allowed interaction protocols, which in turn are defined by means of *Social Integrity Constraints* (ICs). It is a flexible and open framework, in which no assumptions are made on the architecture of the interacting agents, and whatever is not specifically prohibited is allowed.

The society knowledge is defined as an abductive logic program [10]: ICs are used in order to express constraints on the communication patterns, while expected communicative acts ("*expectations*") are expressed as abducible predicates. Both the specification language and the underlying proof-procedure are called $\mathcal{S}$CIFF.

Expectations, whose intuition recalls the usual deontic operators of permission, obligation, and prohibition [9], are used to provide a semantics to both agent communication languages and to interaction protocols [7]. The resulting model is based on a declarative (logic) representation, therefore easy to understand. Moreover, its operational model can be exploited to achieve an implementation of societies of agents based on their formal specifications [3]. Thanks to the link between formal specification and implementation, the model also provides a good ground for the automatic verification and formal proof of properties [11].

The society model and the $\mathcal{S}$CIFF operational model were satisfactorily tested on a number of applications. These include resource exchange [12], e-commerce protocols [8], combinatorial auctions [1], legal contracts [2], and healthcare clinical guidelines [19, 14]. A repository of protocols specified using $\mathcal{S}$CIFF is publicly available through the project's home page [32].

The SOCS-SI tool [5] supports $\mathcal{S}$CIFF models and has been used for extensive experimentation. It reasons from a declarative formalisation of agent interaction protocols, and it offers automated verification procedures to reason about the socially relevant aspects of a SOCS application. SOCS-SI can be adapted to a variety of application domains. It has been interfaced with other implemented agent platforms, such as JADE, and with other non-agent related communication platforms, such as TuCSoN. It implements a version of the $\mathcal{S}$CIFF proof procedure, written in SICStus Prolog, using its CHR library. The interested reader can learn more about $\mathcal{S}$CIFF in [6], and in the tutorial paper [18]. Additional documents and software can be downloaded from the $\mathcal{S}$CIFF and SOCS-SI Web sites [30, 31].

3. Current research directions

Most of our current research originates from the outcomes of SOCS. Starting from the many analogies between the agent paradigm and the Web service model, interaction protocols and choreographies have been the subject of conspicuous research carried out in the context of two recent national projects lead by Alberto Martelli.[2] Part of the research activity carried out within these projects built on $\mathcal{S}$CIFF to produce new formalisms for the specification and verification of interaction protocols and choreographies, and to develop new techniques for automatic property verification and reasoning about Web Services.

We also studied the problem of making the formal languages developed in these projects more easily accessible to a broader audience with the help of graphical notations. We proposed methods to translate choreographies (represented in WS-CDL [23], or in BPMN [29]) into their corresponding $\mathcal{S}$CIFF specification, focussing on verification of compliance. Several tools, based on the $\mathcal{S}$CIFF procedure, have been developed to cope with complete logs and with run-time events. Further supported types of verification regard the proof of "high level" properties. For example, in

[1] Project SOCS, IST-2001-32530, 5FP. "Societies Of ComputeeS: a computational logic model for the description, analysis and verification of global and open societies of heterogeneous computees." See [13, 34].

[2] In 2004–2005, our group was a partner in the National MIUR PRIN project on "Development and verification of logic-based multi-agent systems" [24], and in 2006–2007 on the National MIUR PRIN project on "Specification and verification of agent interaction protocols" [33].

an e-commerce scenario, $\mathcal{S}$CIFF can prove whether a protocol guarantees that a buyer will receive the goods he/she paid for, and that the seller will be paid, by a certain deadline.

We have extended and applied $\mathcal{S}$CIFF in the context of agent-oriented requirements engineering. This has brought to the development of $\mathcal{B}$-Tropos ($\mathcal{B}$ standing for *Business*), a unified framework for information systems engineering, with the aim to reconcile requirements elicitation with declarative specification, prototyping, and analysis [16, 28]. $\mathcal{B}$-Tropos is an extension of the well-known Tropos methodology for agent-oriented software engineering [15]. Thanks to $\mathcal{B}$-Tropos, the can user express temporal and data constraints between tasks, specify start and completion times, triggering events, and deadlines. The $\mathcal{S}$CIFF verification capabilities allow prototyping (animation) and analysis (properties and conformance verification) directly in $\mathcal{B}$-Tropos. This solution offers many advantages. Early requirements engineers will test their models directly. Engineers testing model properties will not have to resort to *ad-hoc*, error-prone translations of high-level models into other languages, thanks to the automatic translation of $\mathcal{B}$-Tropos models into $\mathcal{S}$CIFF programs. Finally, managers monitoring the correct behavior of a system will check the compliance using the SOCS-SI/$\mathcal{S}$CIFF runtime and off-line checking facilities [5].

Another current research direction which builds on SCIFF concerns argumentation in the Semantic Web [37]. Our work resulted in the development of an operational argumentation framework, called ArgSCIFF, to support dialogic argument exchange between Semantic Web Services. In ArgSCIFF, an agent can interact with a Web Service and reason from the interaction result. The reasoning semantics is an argumentation semantics that views the interaction as a dialogue. The dialogue lets two parties exchange arguments and attack, challenge, and justify them on the basis of their knowledge. This format has the potential to overcome a well-known barrier to human users adoption of IT solutions because it permits interaction including justified answers that can be reasoned about and rebutted. Some future research directions in this domain are discussed in [38].

Finally, we are still investigating the use of the $\mathcal{S}$CIFF framework within the MAS research field. Recently, we have focussed our research activity on investigating the similarities between the SOCS approach and commitments [36]. Following this research line, we present in [20] how it is possible to specify commitments in terms of Event Calculus axioms (EC), implemented on top of the $\mathcal{S}$CIFF framework, and how it is possible to exploit the framework to monitor the commitments status at run-time. In [35] we also discuss temporal aspects linked to the notion of commitments, proposing a formalism for specifying temporal constraints related to the commitment properties and their satisfaction. In [22] instead we are investigating exception mechanisms and diagnosis in the commitment paradigm, when the commitments are violated.

4. CLIMB

We are now developing LP-based techniques for modeling and verifying business processes and choreographies. The reference framework for this work is called CLIMB.[3]

As a specification language, CLIMB adopts an extension of DecSerFlow/Condec, a family of graphical languages for the declarative specification of service/business flows [39]. Graphical models are then automatically mapped onto $\mathcal{S}$CIFF, integrating the best of the two approaches:

- CLIMB models are declarative and open. They do not specify one particular flow of execution, but rather focus on the set of constraints that must be satisfied by interacting entities. Constraints specify either what is mandatory or forbidden during execution.
- Different verification tasks can be applied to CLIMB models by exploiting $\mathcal{S}$CIFF as well as different LP techniques.

In particular, CLIMB exploits $\mathcal{S}$CIFF for carrying out both run-time and a-priori verification tasks. At run-time, $\mathcal{S}$CIFF can be used as an alerting infrastructure capable to perform *compliance checking*, i.e., verifying whether a concrete process execution (or service interaction) complies with the prescribed model (and detecting violations as soon as possible). Verification can also be done a-posteriori, to check already completed execution traces. In this respect, CLIMB rules are used as an intuitive classification criterion which splits analyzed traces into a compliant and non compliant sub-sets. We implemented a plug-in which exploits such a reasoning technique, and integrated it inside the ProM [40] process mining framework.

At static time, the "generative" variant of the $\mathcal{S}$CIFF proof procedure can be exploited to check the consistency of developed models, by detecting the presence

[3] CLIMB stands for "Computational Logic for the verIfication and Modeling of Business processes and choreographies".

of conflicts (which undermine the possibility of executing the model) and by discovering if they contain dead activities (i.e., activities that can never be executed). Such verifications also constitute the basis for determining whether different CLIMB models can be composed without introducing conflicts. This is particularly important in a service-oriented setting, where a choreography can be intended as a contract aiming to make different partners correctly collaborate, and then a set of compatible concrete services implementation must be found to concretely implement the system.

It is worth noting that DecSerFlow/Condec models have an alternative underlying semantics in terms of Linear Temporal Logic formulas, which enable the possibility to apply model checking techniques in order to verify the designed models. In this respect, a research activity focused on more foundational aspects is being carried out, to compare expressivity, complexity and reasoning capabilities of the two frameworks.

A comprehensive presentation of the CLIMB framework is given in Montali's book [26], as well as in [27].

Acknowledgements

Much of the work presented here was done in tight cooperation with the AI group of the University of Ferrara. This paper is complementary to [13], where they focus on the learning and property verification issues in relation with the work carried out within and following SOCS.

References

[1] M. Alberti, F. Chesani, M. Gavanelli, A. Guerri, E. Lamma, P. Mello and P. Torroni, Expressing interaction in combinatorial auction through social integrity constraints, *IA* **II**(1) (2005), 22–29.

[2] M. Alberti, F. Chesani, M. Gavanelli, E. Lamma, P. Mello, M. Montali and P. Torroni, Expressing and verifying business contracts with abductive logic programming, *Electronic Commerce, Special Issue on Contract Architectures and Languages* **12**(4) (2008), 9–38.

[3] M. Alberti, F. Chesani, M. Gavanelli, E. Lamma, P. Mello and P. Torroni, A logic based approach to interaction design in open multi-agent systems, in: *Proc 13th IEEE Intl Works On Enabling Technologies: Infrastructures for Collaborative Enterprises (WET-ICE 2004)*, IEEE Press, 2004, pp. 387–392.

[4] M. Alberti, F. Chesani, M. Gavanelli, E. Lamma, P. Mello and P. Torroni, The SOCS computational logic approach for the specification and verification of agent societies, in: *Global Computing*, Springer-Verlag, Berlin, Heidelberg, volume 3267 of *LNAI*, 2005, pp. 324–339.

[5] M. Alberti, F. Chesani, M. Gavanelli, E. Lamma, P. Mello and P. Torroni, Compliance verification of agent interaction: a logic-based tool, *Applied Artificial Intelligence* **20**(2–4) (Feb.–Apr. 2006), 133–157.

[6] M. Alberti, F. Chesani, M. Gavanelli, E. Lamma, P. Mello and P. Torroni, Verifiable agent interaction in abductive logic programming: the SCIFF framework, *ACM Transactions on Computational Logic* **9**(4) (2004) 94–116.

[7] M. Alberti, A. Ciampolini, M. Gavanelli, E. Lamma, P. Mello and P. Torroni, A social ACL semantics by deontic constraints, in: *Multi-Agent Systems and Applications III*, Springer-Verlag, Berlin, Heidelberg, volume 2691 of *LNAI*, 2003, pp. 204–213.

[8] M. Alberti, D. Daolio, P. Torroni, M. Gavanelli, E. Lamma and P. Mello, Specification and verification of agent interaction protocols in a logic-based system, in: *Proceedings of the 19th Annual ACM Symposium on Applied Computing (SAC 2004)*, ACM Press, New York, NY, USA, 2004, pp. 72–78.

[9] M. Alberti, M. Gavanelli, E. Lamma, P. Mello, G. Sartor and P. Torroni, Mapping deontic operators to abductive expectations, *Computational and Mathematical Organization Theory*, **12**(2–3) (Oct. 2006), 205–225.

[10] M. Alberti, M. Gavanelli, E. Lamma, P. Mello and P. Torroni, An abductive interpretation for open societies, in: *Advances in Artificial Intelligence*, Springer-Verlag, Berlin, Heidelberg, volume 2829 of *LNAI*, 2003, pp. 287–299.

[11] M. Alberti, M. Gavanelli, E. Lamma, P. Mello and P. Torroni, Specification and verification of agent interactions using social integrity constraints, *ENTCS* **85**(2) (2004) 94–116.

[12] M. Alberti, M. Gavanelli, E. Lamma, P. Mello and P. Torroni, Modeling interactions using *Social Integrity Constraints*: A resource sharing case study, in: *Declarative Agent Languages and Technologies*, Springer-Verlag, Berlin, Heidelberg, volume 2990 of *LNAI*, May 2004, pp. 243–262.

[13] M. Alberti, M. Gavanelli, E. Lamma, F. Riguzzi and S. Storari, Inducing specification of interacting systems and proving their properties: An approach grounded on computational logic, *IA* **2**(2) (2011), In this issue.

[14] A. Bottrighi, F. Chesani, P. Mello, G. Molino, M. Montali, S. Montani, S. Storari, P. Terenziani and M. Torchio, A hybrid approach to clinical guideline and to basic medical knowledge conformance, in: *12th Conference on Artificial Intelligence in Medicine (AIME'09)*, C. Combi, Y. Shahar and A. Abu-Hanna, eds., volume 5651 of *Lecture Notes in Computer Science*, 2009, pp. 91–95.

[15] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos and A. Perini, Tropos: An agent-oriented software development methodology, *Autonomous Agents and Multi-Agent Systems*, **8** (2004), 203–236.

[16] V. Bryl, P. Mello, M. Montali, P. Torroni and N. Zannone, B-tropos: Agent-oriented requirements engineering meets computational logic for declarative business process modeling and verification, in: *Computational Logic in Multi-Agent Systems VIII*, LNAI, Springer-Verlag, Berlin, Heidelberg, 2008.

[17] M. Bugliesi, E. Lamma and P. Mello, Modularity in logic programming, *Journal of Logic Programming* **19/20** (May/June 1994), 43–502. Special Issue on "10 years of Logic Programming".

[18] F. Chesani, M. Gavanelli, M. Alberti, E. Lamma, P. Mello and P. Torroni, Specification and verification of agent interaction using abductive reasoning (tutorial paper), in: *Computational Logic in Multi-Agent Systems VI*, Springer-Verlag, Berlin, Heidelberg, volume 3900 of *LNAI*, 2006, pp. 243–264.

[19] F. Chesani, P. Mello, M. Montali and S. Storari, Testing careflow process execution conformance by translating a graphical language to computational logic, in: *Proceedings of the 11th International Conference on Artificial Intelligence in Medicine (AIME'07)*, R. Bellazzi, A. Abu-Hanna and J. Hunter, eds., Springer-Verlag, Berlin, Heidelberg, volume 4594 of *Lecture Notes in Computer Science*, 2007, pp. 479–488.

[20] F. Chesani, P. Mello, M. Montali and P. Torroni, Commitment tracking via the reactive event calculus, in: *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009)*, C. Boutilier, ed., 2009, pp. 91–96.

[21] M. Fisher, R.H. Bordini, B. Hirsch and P. Torroni, Computational logics and agents: a road map of current technologies and future trends, *Computational Intelligence* **23**(1) (2007), 61–91.

[22] Ö. Kafalı, F. Chesani and P. Torroni, What happened to my commitment? exception diagnosis among misalignment and misbehavior, in: *Computational Logic in Multi-Agent Systems*, J. Dix, J. Leite, G. Governatori and W. Jamroga, eds., Springer Berlin/Heidelberg, volume 6245 of *Lecture Notes in Computer Science*, 2010. pp. 82–98.

[23] N. Kavantzas, D. Burdett, G. Ritzinger, T. Fletcher and Y. Lafon, Web Services Choreography Description Language, Version 1.0, W3C Working Draft 17-12-04, 2004.

[24] MASSiVE: sviluppo e verifica di sistemi multi-agente basati sulla logica, http://www.di.unito.it/massive.

[25] P. Mello, P. Torroni, M. Gavanelli, M. Alberti, A. Ciampolini, M. Milano, A. Roli, E. Lamma, F. Riguzzi and N. Maudet, A logic-based approach to model interaction amongst computees, Technical report, SOCS Consortium, 2003. Deliverable D5. Available from the SOCS project web site [32].

[26] M. Montali, Specification and verification of declarative open interaction models, *Lecture Notes in Business Information Processing*, Springer-Verlag GmbH, Berlin and Heidelberg, Germany, Vol. 56, 2010.

[27] M. Montali, M. Pesic, W. van der Aalst, F. Chesani, P. Mello and S. Storari, Declarative specification and verification of service choreographies, *ACM Transactions on the Web* **4**(1) (2010).

[28] M. Montali, P. Torroni, N. Zannone, P. Mello and V. Bryl, Engineering and verifying agent-oriented requirements augmented by business constraints with B-Tropos, *Autonomous Agents and Multi-Agent Systems* (2010), 1–31. doi:10.1007/s10458-010-9135-4.

[29] Object Management Group/Business Process Management Initiative, *Business Process Modeling Notation*, 2007. Home Page, http://www.bpmn.org.

[30] The $\mathcal{SCIFF}$ Abductive Proof Procedure, 2005–2010. http://lia.deis.unibo.it/research/sciff.

[31] SOCS-$\mathcal{SI}$ Home Page, 2006. http://lia.deis.unibo.it/research/socs_si.

[32] Societies Of ComputeeS (SOCS): a computational logic model for the description, analysis and verification of global and open societies of heterogeneous computees. IST-2001-32530, 2002-2005. http://lia.deis.unibo.it/research/socs.

[33] Specifica e verifica di protocolli di interazione fra agenti. http://www.di.unito.it/svp.

[34] F. Toni, Multi-agent systems in computational logic: Challenges and outcomes of the SOCS project, in: *Computational Logic in Multi-Agent Systems VI*, Springer-Verlag, Berlin, Heidelberg, volume 3900 of *LNAI*, 2006, pp. 420–426.

[35] P. Torroni, F. Chesani, P. Mello and M. Montali, Social commitments in time: Satisfied or compensated, in: *Declarative Agent Languages and Technologies VII (DALT 2009). Revised Selected and Invited Papers*, M. Baldoni, J. Bentahar, M.B. van Riemsdijk and J. Lloyd, eds., Springer-Verlag, Berlin, Heidelberg, volume 5948 of *Lecture Notes in Computer Science*, 2010, pp. 228–243.

[36] P. Torroni, P. Yolum, M. P. Singh, M. Alberti, F. Chesani, M. Gavanelli, E. Lamma, and P. Mello,*Modelling interactions via commitments and expectations*. chapter in: *Multi-Agent Systems: Semantics and Dynamics of Organizational Models*, Virginia Dignum, eds., IGI Global, Hershey, Pennsylvania (USA), 2009, pp. 263–284. ISBN: 978-1-60566-256-5.

[37] P. Torroni, M. Gavanelli and F. Chesani, Argumentation in the semantic web, *IEEE Intelligent Systems* **22**(6) (Nov./Dec. 2007), 66–74.

[38] P. Torroni, M. Gavanelli and F. Chesani, Arguing on the semantic grid, in: *Argumentation in AI*, I. Rahwan and G. Simari, eds., Springer-Verlag, Berlin, Heidelberg, 2009, pp. 423–441.

[39] W.M.P. van der Aalst and M. Pesic, Decserflow: Towards a truly declarative service flow language, in: *Proceedings of the Third International Workshop on Web Services and Formal Methods (WS-FM 06)*, M. Bravetti, M. Núñez and G. Zavattaro, eds, Springer-Verlag, Berlin, Heidelberg, volume 4184 of *LNCS*, 2006, pp. 1–23.

[40] W.M.P. van der Aalst, B.F. van Dongen, C.W. Günther, R.S. Mans, A.A. de Medeiros, A. Rozinat, V. Rubin, M. Song, H.M.W. Verbeek and A.J.M.M. Weijters, ProM 4.0: Comprehensive support for real process analysis, in: *Application and Theory of Petri Nets and Other Models of Concurrency (ICATPN 2007)*, J. Kleijn and A. Yakovlev, eds., Springer-Verlag, Berlin, Heidelberg, volume 4546 of *LNCS*, 2007, pp. 484–494.