Engineering Business Processes through Accountability and Agents

Extended Abstract

Matteo Baldoni
Università di Torino, Dipartimento di Informatica, Italy

Cristina Baroglio
Università di Torino, Dipartimento di Informatica, Italy

Olivier Boissier
Laboratoire Hubert Curien UMR CNRS 5516, Institut Henri Fayol, MINES Saint-Etienne, France

Roberto Micalizio
Università di Torino, Dipartimento di Informatica, Italy

Stefano Tedeschi
Università di Torino, Dipartimento di Informatica, Italy

KEYWORDS

Accountability; Responsibility; Multiagent Organizations; JaCaMo.

ACM Reference Format:
Matteo Baldoni, Cristina Baroglio, Olivier Boissier, Roberto Micalizio, and Stefano Tedeschi. 2019. Engineering Business Processes through Accountability and Agents. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 3 pages.

1 RESPONSIBILITY AND ACCOUNTABILITY AS ENGINEERING CONCEPTS

A business process (BP) is "a set of activities that are performed in coordination in an organizational and technical environment. These activities jointly realize a business goal." [26]. In general, a business goal is achieved by breaking it up into sub-goals, which are distributed to a number of actors. Each actor carries out part of the process, and depends on the collaboration of others to perform its task. Multiagent Systems (MAS), in particular models for MAS organizations (MAO), are promising candidates to supply the right abstractions for describing BPs; however agent organizations still lack a systematic way to properly handle feedback of the execution of business processes in terms of good or bad functioning (e.g. *exceptions*). When a fedback occurs, the agent which can handle it (or which is interested to know), may be not the same agent who triggers the feedback. To make the overall system robust, the feedback should be reported to the agent with the proper means for treating it. In [2] a proposal was made to use accountability and responsibility relationships to state the rights and duties of agents in the organisation, given the specification of a normative organization. Building upon this work, we have studied how *robustness* can be achieved via accountability and responsibility relationships, and we used these concepts as tools to systematize and guide the design and development of the agents. It is worth noting that accountability and responsibility are not primitive concepts. Rather, they are properties that emerge in carefully designed software systems. Thus using them as engineering tools means that we actually constrain the ways in which software is designed and developed.

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

We then exemplified how such concepts can be engineered in a JaCaMo MAO, where agents execute under a normative organization expressing BPs as accountability and responsibility relations among agents. Specifically, we considered the OMG Incident Management scenario [21]. The case models the interaction between a customer and a company for the management of a problem reported by the customer. The customer reports the problem to a Key Account Manager who, on the basis of her experience, can either resolve the problem directly or ask for the intervention of first-level support. The problem is, then, recursively treated at different support levels until, in the worst case, it is reported to the software developer. Generally, the business aim of the process (to solve the reported problem) is decomposed and can be distributed over five BPMN processes, whose execution requires their interaction and coordination. Noticeably, as always with BPs, the way in which goals are achieved matters, so agents are expected not only to fulfill their assigned goals but also to respect the BP: the "goal" is that the process takes place [1].

Goal distribution over a group of BPs bears strong similarities with proposals about MAO. For what concerns software modularity, both suffer from some limitations. By focussing merely on the achievement of the assigned sub-goals, agents loose sight of the overall process, and *ignore the place* their achievement has within the organization. The relationship between each level of support and the following one, in the example, is emblematic: when a request of support is made, an answer containing some kind of feedback on the realisation of this support is expected in order to proceed. However, since processes are independent, one cannot give for granted that another will answer. It follows that when a process does not answer, the waiting one may get stuck indefinitely. Similarly, MAOs (e.g., [11, 14]) allow the functional decomposition of complex, organizational goals, and the assignement of subgoals to agents. The coordinated execution of subgoals is often supported by a normative specification, by which the organization issues obligations towards the agents, e.g. [8, 12, 13, 16]. However, agents may operate in ways that do not fit into the process specification and, importantly, when agents fail, the organization has no explicit mechanism for sorting out what occurred, for a redress.

This is where accountability [3–6, 10, 17, 18] and responsibility [5, 23, 25, 27] come in handy. Accountability "emerges as a primary characteristic of governance where there is a sense of agreement

and certainty about the legitimacy of expectations between the community members." [15]. In particular [18], accountability implies that some actors have the right to hold other actors to a set of standards, to judge whether they have fulfilled their responsibilities in light of these standards, and to impose sanctions if they determine that these responsibilities have not been met. It presupposes a relationship between power-wielders and those holding them accountable, where there is a general recognition of the legitimacy of (1) the operative standards for accountability and (2) the authority of the parties to the relationship (one to exercise particular powers and the other to hold them to account). BPs represent an agreed behavior, introduce expectations on the behavior of the interacting parties, and require some kind of governance in order for the process to be enacted, but the accountability results hidden into some kind of collective responsibility ("many hands problem"). As a consequence, the governance of the system is compromised as well as its functioning as a whole. This is what we aim at overcoming.

2 ENGINEERING MAO WITH ACCOUNTABILITY/RESPONSIBILITY

Since accountabilities and responsibilities imply some obligations [18], we can realize them in JaCaMo by relying on the deontic primitives that such framework provides. In other words, the fitting projection over role x (see [2]) can be mapped into a number of Jason plans of the agent playing role x by way of the following patterns, expressed in AgentSpeak(ER). Specifically, the fitting relationship represented by each pair $\langle R(x,q), A(x,y,r,u)\rangle$ in $\mathbf{R}_x \rightsquigarrow \mathbf{A}_x$, is mapped into an AgentSpeak(ER) g-plan as follows:

+!be_accountable(x, y, q) <: drop_fitting(x, y, q) {
// Well-Doing e-plan
+obligation(x, q) : $r \wedge c$ <- $body_q$.
// Wrong-Doing e-plan
+oblUnfulfilled(x, q) : $r \wedge c'$ <- $body_f$. }

Such that: (1) $body_q$ satisfies the *fitting-adherence* condition (see below); (2) $body_f$ includes the sending of an explanation for the failure from x to y. The two e-plans encode the proactive behavior of an agent assuming a responsibility. Until the responsibility is not dropped, the agent starts reacting to obligations in accordance to the accountability specified in the fitting. The agent will perceive, through the identity that is provided by the organizational role it plays, certain events as events it should tackle through some behavior of its own, but it will also be aware of its social position both (1) by knowing some other agent will have the right, under certain conditions, to ask for an account and (2) by including specific behavior for building such an account.

Well-doing e-plan: is triggered when the specified obligation is issued by the MAO. The context expression, $r \wedge c$, is satisfied when condition r activating the agent accountability holds together with some possibly empty local condition c to choose among alternative plans, i.e., multiple ways to achieve a same result in different (local) circumstances. Due to the accountability fitting the agent has accepted, the body of the plan(s) ($body_q$) must, then, be such to satisfy the responsibility assumption represented by the pair $\langle R(x,q), A(x,y,r,u)\rangle$. That is, the plan body has to satisfy a *fitting-adherence* condition, by which there must exist an execution of the plan body that, restricted to the events that are relevant for the progression of u, is an actualization of the responsibility q. Intuitively, q is actually used for fulfilling the obligation. In this case, the obligation to give an account for the satisfaction of the obligation is implicitly resolved by satisfying the very same obligation. It is interesting to note that the accountability fitting is not only a functional specification of the organization, but it also specifies the "good" behavior of the agents. It is in fact this characteristic that justifies our programming patterns, that enriches the standard JaCaMo [9].

Wrong-doing e-plan: allows the agent to provide an account when it did not complete its task. The triggering event, *oblUnfulfilled*, is generated by the MAO when an obligation is left unsatisfied. The context has the same structure as above; $body_f$ produces an account of the failure. This will be an explanation that the agent produces and that some other agent will use to manage the exception and to resume the execution. The correct use of the pattern guarantees, by design, that exceptional events, when occurring, are reported to the agents who can handle them properly. Accountability fulfills this purpose because, by nature, it brings about an obligation on the a-giver to give an account of what it does.

The engineering of accountable JaCaMo MAOs involves that: (1) Each process is mapped to an organizational role; (2) A scheme representing the overall process goal is defined (its successful execution corresponds to the achievement of the process goal; (3) For each activity to be performed in sequence, a subgoal is added to the scheme by means of the corresponding operator; (4) For each structured block including a concurrent execution, the corresponding goals, grouped by the parallel operator, are added to the scheme; (5) for each choice all the schemes representing the possible courses of action should be defined. They will be instantiated dynamically by the agents, depending on their internal choices.

3 CONCLUSIONS

The systematic application of the proposed patterns makes agents aware of the process as characterization of the goal. Hence, accountabilities provide the programmer with a behavioral specification the agent has to satisfy. The proposal moves MAOs closer to other paradigms where exceptions are handled. In the actor model [19], for instance, when an actor cannot handle an exception, it reports the exception to its parent actor, which decides to either handle the exception or report it further up. In an agent-based system such a scheme is not directly applicable because agents are independent entities, and show no parent-child relationship. Approaches for modeling exceptions in a MAS setting have been proposed (see, e.g., [20, 22, 24]). However, no consensus has been reached on the use of such a concept in agent systems. The main problems rise when trying to accomodate the usual exception handling semantics with the properties of MAS; namely autonomy, openness, heterogeneity, and encapsulation. Accountabilities can fill this gap.

Commitment-based protocols, e.g. [28], as well as NorMAS [7], provide alternatives for modeling coordination. A detached commitment is an obligation on the debtor to bring about the consequent. In case of violation, the creditor has the right to complain against the debtor but cannot hold the debtor to provide an explanation. This lack of information hampers both the understanding of what has occurred, and attempts of recovery from the failure.

REFERENCES

[1] Greta Adamo, Stefano Borgo, Chiara Di Francescomarino, Chiara Ghidini, and Nicola Guarino. 2018. On the Notion of Goal in Business Process Models. In *AI*IA 2018 - Advances in Artificial Intelligence - XVIIth International Conference of the Italian Association for Artificial Intelligence, Trento, Italy, November 20-23, 2018, Proceedings (Lecture Notes in Computer Science)*, Chiara Ghidini, Bernardo Magnini, Andrea Passerini, and Paolo Traverso (Eds.), Vol. 11298. Springer, 139–151. https://doi.org/10.1007/978-3-030-03840-3_11

[2] Matteo Baldoni, Cristina Baroglio, Olivier Boissier, Katherine M. May, Roberto Micalizio, and Stefano Tedeschi. 2018. Accountability and Responsibility in Agents Organizations. In *PRIMA 2018: Principles and Practice of Multi-Agent Systems, 21st International Conference (Lecture Notes in Computer Science)*, T. Miller, N. Oren, Y. Sakurai, I. Noda, T. Savarimuthu, and Tran Cao Son (Eds.). Springer, Tokyo, Japan, 403–419. http://dx.doi.org/10.1007/978-3-030-03098-8_16

[3] Matteo Baldoni, Cristina Baroglio, Katherine M. May, Roberto Micalizio, and Stefano Tedeschi. 2016. Computational Accountability. In *Deep Understanding and Reasoning: A challenge for Next-generation Intelligent Agents, URANIA 2016*, F. Chesani, P. Mello, and M. Milano (Eds.), Vol. 1802. CEUR, Workshop Proceedings, Genoa, Italy, 56–62. http://ceur-ws.org/Vol-1802/paper8.pdf

[4] Matteo Baldoni, Cristina Baroglio, Katherine M. May, Roberto Micalizio, and Stefano Tedeschi. 2018. An Information Model for Computing Accountabilities. In *AI*IA 2018: Advances in Artificial Intelligence, XVII International Conference of the Italian Association for Artificial Intelligence (Lecture Notes in Computer Science)*, C. Ghidini, B. Magnini, A. Passerini, and P. Traverso (Eds.), Vol. 11298. Springer, Trento, Italy, 30–44. https://doi.org/10.1007/978-3-030-03840-3_3/

[5] Matteo Baldoni, Cristina Baroglio, Katherine M. May, Roberto Micalizio, and Stefano Tedeschi. 2018. Computational Accountability in MAS Organizations with ADOPT. *Applied Sciences* 8, 4 (2018).

[6] Matteo Baldoni, Cristina Baroglio, and Roberto Micalizio. 2018. Goal Distribution in Business Process Models. In *AI*IA 2018: Advances in Artificial Intelligence, XVII International Conference of the Italian Association for Artificial Intelligence (Lecture Notes in Computer Science)*, C. Ghidini, B. Magnini, A. Passerini, and P. Traverso (Eds.), Vol. 11298. Springer, Trento, Italy, 252–265. https://doi.org/10.1007/978-3-030-03840-3_19

[7] Guido Boella, Leendert W. N. van der Torre, and Harko Verhagen. 2007. Introduction to Normative Multiagent Systems. In *Normative Multi-agent Systems (Dagstuhl Seminar Proceedings)*, Vol. 07122.

[8] Olivier Boissier, Rafael H. Bordini, Jomi F. Hübner, Alessandro Ricci, and Andrea Santi. 2013. Multi-agent Oriented Programming with JaCaMo. *Sci. Comput. Program.* 78, 6 (2013), 747–761. https://doi.org/10.1016/j.scico.2011.10.004

[9] Olivier Boissier, Rafael H. Bordini, Jomi F. Hübner, Alessandro Ricci, and Andrea Santi. 2013. Multi-agent oriented programming with JaCaMo. *Science of Computer Programming* 78, 6 (2013), 747 – 761. https://doi.org/10.1016/j.scico.2011.10.004

[10] Amit K Chopra and Munindar P Singh. 2016. From social machines to social protocols: Software engineering foundations for sociotechnical systems. In *Proc. of the 25th Int. Conf. on WWW*.

[11] Daniel D. Corkill and Victor R. Lesser. 1983. The Use of Meta-Level Control for Coordination in Distributed P roblem Solving Network. In *Proceedings of the 8th International Joint Conference on Ar tificial Intelligence (IJCAI'83)*, Alan Bundy (Ed.). William Kaufmann, Los Altos, CA, 748–756.

[12] Mehdi Dastani, Nick AM Tinnemeier, and John-Jules Ch Meyer. 2009. A programming language for normative multi-agent systems. In *Handbook of Research on Multi-Agent Systems: semantics and dynamics of organizational models*. IGI Global, 397–417.

[13] Virginia Dignum. 2004. *A model for organizational interaction: based on agents, founded in logic.* Ph.D. Dissertation. Utrecht University. Published by SIKS.

[14] Virginia Dignum. 2009. Handbook of Research on Multi-agent Systems: Semantics and Dynamics of Organizational Models. (2009).

[15] Melvin J. Dubnick and Jonathan B. Justice. 2004. Accounting for Accountability. (September 2004). https://pdfs.semanticscholar.org/b204/36ed2c186568612f99cb8383711c554e7c70.pdf Annual Meeting of the American Political Science Association.

[16] Nicoletta Fornara, Francesco Viganò, Mario Verdicchio, and Marco Colombetti. 2008. Artificial institutions: a model of institutional reality for open multiagent systems. *Artificial Intelligence and Law* 16, 1 (2008), 89–105. https://doi.org/10.1007/s10506-007-9055-z

[17] Harold Garfinkel. 1967. *Studies in ethnomethodology.* Prentice-Hall Inc., Englewood Cliffs, New Jersey.

[18] Ruth W. Grant and Robert O. Keohane. 2005. Accountability and Abuses of Power in World Politics. *The American Political Science Review* 99, 1 (2005).

[19] Philipp Haller and Frank Sommers. 2011. *Actors in Scala - concurrent programming for the multi-core era.* Artima.

[20] Ashok U. Mallya and Munindar P. Singh. 2005. Modeling Exceptions via Commitment Protocols. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS '05).* ACM, 122–129.

[21] Object Management Group. 2018. BPMN Specification - Business Process Model and Notation. (2018). http://www.bpmn.org/ Online, accessed 08/11/2018.

[22] Eric Platon, Nicolas Sabouret, and Shinichi Honiden. 2008. An Architecture for Exception Management in Multiagent Systems. *Int. J. Agent-Oriented Softw. Eng.* 2, 3 (2008), 267–289.

[23] Ian Sommerville, Russell Lock, Tim Storer, and John Dobson. 2009. Deriving Information Requirements from Responsibility Models. In *Advanced Information Systems Engineering, 21st International Conference, CAiSE 2009, Amsterdam, The Netherlands, June 8-12, 2009. Proceedings.* 515–529.

[24] Frédéric Souchon, Christophe Dony, Christelle Urtado, and Sylvain Vauttier. 2004. Improving Exception Handling in Multi-agent Systems. In *Software Engineering for Multi-Agent Systems II.* Springer Berlin Heidelberg, 167–188.

[25] Nicole A. Vincent. 2011. *Moral Responsibility.* Library of Ethics and Applied Philosophy, Vol. 27. Springer, Chapter A Structured Taxonomy of Responsibility Concepts.

[26] Mathias Weske. 2007. *Business Process Management: Concepts, Languages, Architectures.* Springer.

[27] Vahid Yazdanpanah and Mehdi Dastani. 2016. Distant Group Responsibility in Multi-agent Systems. In *PRIMA 2016: Princiles and Practice of Multi-Agent Systems - 19th International Conference, Phuket, Thailand, August 22-26, 2016, Proceedings.* 261–278. https://doi.org/10.1007/978-3-319-44832-9_16

[28] P. Yolum and M. P. Singh. 2002. Commitment Machines. In *Intelligent Agents VIII, 8th Int. WS, ATAL 2001 (LNCS)*, Vol. 2333. Springer, 235–247.