Exploiting Social Commitments in Programming Agent Interaction

Matteo Baldoni, Cristina Baroglio, Federico Capuzzimati, Roberto Micalizio

Università degli Studi di Torino — Dipartimento di Informatica

Abstract. Modeling and regulating interactions among agents is a critical step in the development of Multiagent Systems (MASs). Some recent works assume a normative view, and suggest to model interaction protocols in terms of obligations. In this paper we propose to model interaction protocols in terms of goals and commitments, and show how such a formalization promotes a deliberative process inside the agents. The proposal is implemented via JaCaMo+, an extension of JaCaMo, in which Jason agents can interact, while preserving their deliberative capabilities, by exploiting commitment-based protocols, reified by special CArtAgO artifacts.

Keywords: Social Computing, Agent Programming, Commitments and Goals, Agents & Artifacts, JaCaMo

1 Introduction

Many researchers claim that an effective way to approach the design and development of a MAS consists in conceiving it as a structure composed of four main entities: *Agents*, *Environment*, *Interactions*, and *Organization* [13,14,6]. Such a separation of concerns enjoys many advantages from a software engineering point of view, since it enables a modular development of code that eases code reuse and maintainability. Currently, there are many frameworks that support the realization of one of these components (e.g., [4,8]). To the best of our knowledge, JaCaMo [5] is the the most complete among the well-established proposals, providing a thorough integration of the three components agents, environments, and organizations into a single programming framework.

A recent extension to JaCaMo [25] further enriches the framework by introducing an interaction component. The interaction component allows regulating both agent interactions and the interactions between agents and environment. More precisely, an interaction component encodes –in an automaton-like shape– a *protocol*, in which states represent protocol steps, and transitions between states are associated with (undirected) *obligations*. Such protocols provide a *guideline* of how a given organizational goal should be achieved. Interaction components, as defined in [25], however, present also some drawbacks. Works such as [11] show the importance, for the agents to be autonomous, to reason about the social consequences of their actions by exploiting *constitutive norms* that link the agents' actions to their respective social meanings. However, an interaction component operates as a coordinator that, by relying on obligations, issues commands about what an agent has to do, and when. This impedes agents from reasoning on the normative effects of their actions. On the one hand, obligations are not constitutive norms, on the other hand, the social meaning of such commands is not known to the agents but only implicitly encoded within the protocol. Agents lose part of their deliberative power since, once they join an interaction component, they have no other choice but deciding whether satisfying or not those obligations they are in charge of, while the rationale behind these obligations remains hidden to them. Consequently, this approach does not suit those situations where interaction is not subject to an organizational guideline, such as when interaction is among agents and each agent decides what is best for itself [24], or when guidelines amount to declarative, underspecified constraints that still leave agents the freedom to take strategic decisions on their behavior. We thus propose a complementary approach to [25] which better supports the deliberative capabilities of the agents.

When organizational goals are not associated with corresponding guidelines, agent deliberation is crucial for the achievement of goals. An agent has to act not only upon its own goals, but also upon what interactions could be necessary for achieving these goals. In other terms, an agent has to discover how to fulfill a goal by interacting with others. It is important to underline that when agents can fully exploit their deliberative capabilities, they can take advantage of opportunities (*flexibility*), and can find alternative ways to get their goals despite unexpected situations that may arise (*robustness*). To this aim, we present JaCaMo+, an agent platform, that builds upon JaCaMo [5], where Jason agents engage *commitment-based interactions* which are reified as CArtAgO artifacts. CArtAgO is a framework based on the A&A meta-model [23,19] which extends the agent programming paradigm with the first-class entity of artifact: a resource that an agent can use, and that models working environments. The environment is itself programmable and encapsulates services and functionalities, making it active. JaCaMo+ artifacts represent the *interaction social state* and provide the roles agents enact. The use of artifacts enables the implementation of monitoring functionalities for verifying that the on-going interactions respect the commitments and for detecting violations and violators. The well-known gold miners scenario is used as an example.

2 Social commitments for programming

The heart of our proposal is that whenever organization-driven guidelines are missing, the interactions among the agents should be supported by the very fundamental notions of *goal* and *engagement*. So, we propose to complement the interaction protocol in [25], and more in general organizational and normative approaches [12,15,18], with an *interaction artifact* that can be used by the agents as a common ground. Our interaction artifacts encode the notion of engagement as *social commitment* [20]. A social commitment models the directed relation between two agents: a *debtor* and a *creditor*, that are both aware of the existence of such a relation and of its current state: A commitment $\mathsf{C}(x, y, s, u)$ captures that agent x (debtor) commits to agent y (creditor) to bring about the consequent condition u when the antecedent condition s holds. Antecedent and consequent conditions are conjunctions or disjunctions of events and commitments. A commitment is autonomously taken by a debtor towards a creditor on its own initiative, instead of dropping from an organization, like obligations. Unlike obligations, commitments are manipulated by agents through the standard operations *create*, *cancel*, *release*, *discharge*, *assign*, *delegate* [20]. Since debtors are expected to satisfy their engagements, commitments have a normative value, providing social expectations on the agents' behaviors, as well as obligations.

The choice of commitments is, thus, motivated by the fact that they are taken by an agent as a result of an internal deliberative process, that creates social relationships with a normative flavour. This preserves the autonomy of the agents and is fundamental to harmonize deliberation with goal achievement. The agent does not just react to some obligations, but it rather includes a deliberative capacity by which it creates engagements towards other agents while it is trying to achieve its goals (or to the aim of achieving its goals). Citing Singh [21], an agent would become a debtor of a commitment based on the agent's own communications: either by directly saying something or having another agent communicate something in conjunction with a prior communication of the debtor. That is, there is a causal path from the establishment of a commitment to prior communications by the debtor of that commitment. By contrast, obligations can result from a deliberative process which is outside the agent; this is the case of the interaction component in [25]. This is the reason why we believe that the introduction of a deliberative process on constitutive rules that rely on obligations would not really support the agents' autonomy.

Commitment-based protocols assume that a (notional) *social state* is available and inspectable by all the involved agents. The social state traces which commitments currently exist between any two agents, and the states of these commitments according to the commitments lifecycle. Commitments can be used by agents in their practical reasoning together with beliefs, intentions, and *goals*. In particular, Telang et al. [22] point out that goals and commitments are one another complementary: A commitment specifies how an agent relates to another one, and hence describes what an agent is willing to bring about for another agent. On the other hand, a goal denotes an agent's proattitude towards some condition; that is, a state of the world that the agent should achieve. An agent can create a commitment towards another agent to achieve one of its goals; but at the same time, an agent determines the goals to be pursued relying on the commitments it has towards others.

3 JaCaMo+

JaCaMo [5] is a platform integrating Jason (as an agent programming language), CArtAgO (as a realization of the A&A meta-model [23]), and Moise (as a support to the realization of organizations). JaCaMo+ extends the CArtAgO and Jason components of the standard JaCaMo. Artifacts are enriched with an explicit representation of commitments and of commitment-based protocols. The resulting class of artifacts reifies the execution of commitment-based protocols, including the social state of the interaction, and enables Jason agents both to be notified about the social events and to perform practical reasoning also about the other agents (this is possible thanks to the *social expectations* raised by commitments). Specifically, a JaCaMo+ artifact encodes a commitment protocol, that is structured into a set of roles. By enacting a role, an agent gains the rights to perform social actions, whose execution has public social consequences, expressed in terms of commitments. If an agent tries to perform an action which is not associated with the role it is enacting, the artifact raises an exception that is notified to the violator. Instead, when an agent performs a protocol action that pertains to its role, the social state is updated accordingly by adding new commitments, or by modifying the state of existing commitments. Since an artifact is a programmable, active entity, it can act as a monitor of the interaction in progress, detecting violations that it can ascribe to the violator without the need of agent introspection. By *focusing* on an artifact, an agent registers to be notified of events that are generated inside the artifact: when the social state is updated, the JaCaMo+ artifact provides such information to the focusing JaCaMo+ agents by exploiting proper *observable properties*. Agents are, thus, constantly aligned with the social state.

Jason [8] implements in Java, and extends, the agent programming language AgentSpeak(L). Jason agents have a BDI architecture. Each has a belief base, and a plan library. It is possible to specify *achievement* (operator '!') and *test* (operator '?') goals. Each plan has a triggering event (causing its activation), which can be either the addition or the deletion of some belief or goal. JaCaMo+ extends JaCaMo by allowing the specification of plans whose triggering events involve commitments. JaCaMo+ represents a commitment as a term *cc*(*debtor*, *creditor*, *antecedent*, *consequent*, *status*) where *debtor* and *creditor* identify the involved agents (or agent roles), while *antecedent* and *consequent* are the commitment conditions. *Status* is the commitment state (the set being defined in the commitments life cycle [16]). Commitment operations are realized as *internal operations* of the new class of artifacts we added to CArtAgO. Thus, they cannot be invoked directly by the agents, but the protocol actions will use them as primitives to modify the social state.

A Jason plan is specified as $triggering\_event : \langle context \rangle \leftarrow \langle body \rangle$. The *triggering_event* denotes the events the plan handles, the *context* specifies the circumstances when the plan could be used, the *body* is the course of action that should be taken. Commitments can be used both in the *context* and in the *body*. Otherwise than beliefs, their assertion/deletion can only occur through the artifact, in consequence to a social state change. The following template shows a Jason plan triggered by the addition of a commitment in the social state: $+cc(debtor, creditor, antecedent, consequent, status) : \langle context \rangle \leftarrow \langle body \rangle$. More precisely, the plan is triggered when a commitment, that unifies with the one in the plan head, appears in the social state. The syntax is the standard for Jason plans. *Debtor* and *creditor* are to be substituted by the proper roles. The plan may be devised so as to change the commitment status (e.g. the debtor will try to satisfy the comment), or it may be devised so as to allow the agent to react to the commitment presence (e.g., collecting information). Similar schemas can be used for commitment deletion and for the addition/deletion of social facts. Further, commitments can also be used in contexts and in plans as test goals ($?cc(\dots)$), or achievement goals ($!cc(\dots)$). Addition or deletion of such goals can, as well, be managed by plans; for example: $+!cc(\mathit{debtor}, \mathit{creditor}, \mathit{antecedent}, \mathit{consequent}, \mathit{status}) : \langle \mathit{context} \rangle \leftarrow \langle \mathit{body} \rangle$. The plan is triggered when the agent creates an achievement goal concerning a commitment. Consequently, the agent will act upon the artifact so as to create the desired social relationship. After the execution of the plan, the commitment $cc(\mathit{debtor}, \mathit{creditor}, \mathit{antecedent}, \mathit{consequent}, \mathit{status})$ will hold in the social state, and will be projected onto the belief bases of all agents focusing on the artifact.

4 JaCaMo+ Gold Miners

The CLIMA VII gold miners scenario consisted of developing a multi-agent system to solve a cooperative task in a dynamically changing environment: a grid-like world where agents could move from one cell to a neighbouring cell if it contained no agent or obstacle. Gold could appear in the cells. Agent teams were expected to explore the environment, avoid obstacles and compete with another agent team for collecting as much gold as they could and deliver it to the depot. Each agent can carry one gold nugget at a time (we say that an agent that is not carrying gold is free). Agents had a local view on environment, their perceptions could be incomplete, and their actions could fail.

```
@goldpercepted[atomic]
+cell(X,Y,gold) : free & not gold(X,Y) & enactment_id(My_Role_Id)
        & not cc(My_Role_Id,_,Assigned,Drop,"DETACHED")
   <-   +gold(X,Y); commitToDropGold(X,Y).
@goldperceptedswitch[atomic]
+cell(X,Y,gold) : not free & not carrying(_,_)  & not gold(X,Y)
        & handling_gold(OldX,OldY) & not changed
   <-   .drop_intention(handle(gold(_,_)));
        +gold(X,Y); +changed; -handling_gold(OldX,OldY);
        changeGoldToPursue(X,Y); communicateGoldPosition(OldX,OldY).
+cell(X,Y,gold) : not gold(X,Y)
   <-   communicateGoldPosition(X,Y).
@commitgoldfree[atomic]
+gold(X,Y) : free & pos(myX, myY) & not bet(X,Y)
   <-   jia.dist(X1,Y1,X2,Y2,Dist); bid(X,Y,Dist); +bet(X,Y).
+gold(X,Y) : not bet(X,Y)
   <-  ignore(X,Y); +bet(X,Y).
@p2gold[atomic]
+cc(My_Role_Id, _, _, drop(X,Y), "DETACHED")
    : enactment_id(My_Role_Id)
  <- -free; !init_handle(gold(X,Y)).
```

Listing 1.1. The gold miner agent code in JaCaMo+.

We used four miners, each executing the same code, part of which is reported in Listing 1.1, the code is at http://di.unito.it/2COMM. The four agents are randomly positioned within the map and start searching for gold. A JaCaMo+ commitment protocol artifact is also created and shared by all the miners. All miners focus on the artifact and, thus, will be notified of changes occurred to its observable properties. This simple mechanism is, for instance, used for handling the case when an agent bumps into gold –*cell(X, Y, gold)*, gold perceived by agent in a certain cell– but, since it is already carrying a gold nugget to the depot, it cannot pick it up (11-12). Then, it communicates its discovery to its team mates, so that someone else can handle the newly found gold. To this aim, it invokes the artifact operation *communicateGoldPosition(X, Y)*, which causes the assertion of an observable property in the social state, which is notified to all agents in their belief bases as the belief *gold(X, Y)*. When the agent that finds the gold is free (2-4), it creates (*commitToDropGold(X, Y)*) a detached commitment towards all other agents, $C(My\_Role\_Id, Others, \top, drop(X, Y))$, of which it is the debtor, to bring the newly found gold to the depot. This will, in turn, activate the plan at lines 19-21 to handle that nugget. If, instead, the agent is not free because when it found the gold it was actually aiming at another nugget, the agent will change its plans (6-10), setting as gold to pick up the newly found nugget, and will communicate, through the artifact, the coordinates of the gold it was previously aiming at, so that someone else can handle it. This is done by the operations *changeGoldToPursue(X, Y)*, which withdraws the commitment to drop the assigned nugget and creates a commitment to drop the just perceived one, and *communicateGoldPosition(OldX, OldY)*.

The appearence of the belief *gold(X, Y)* in an agent's belief base triggers a plan. A free agent (14-15) will execute the artifact operation *bid*, which causes the creation of a *conditional commitment*: if allocated the task, the agent will collect the gold. So, *bid* creates a social engagement, whose debtor is the bidding miner, and the creditor is the whole class of team mates. The agent is requested to include one or more behaviors for managing such a commitment and, in particular, for managing the case in which it is *Detached*, i.e. when the gold nugget is allocated to the agent. This is possible because *bid* and the *commitment* $C(miner_i, all\_miners, allocated(X, Y, miner_i), drop(X, Y))$ are tied by the social meaning of the operation in an explicit way, and this information is available to the programmer who will add to the agent program plans for handling the commitment state changes it needs to handle. Knowing the social meanings of artifact operations is sufficient for coordinating with others correctly. Agents that are not free just ignore the new gold (line 16). It is possible to distinguish the two cases by properly defining the plan context.

Instead, in [7] the relation between *bid* and nugget allocation (the latter is a consequence of the former), that is fundamental to the programmer, is *hidden* inside the the leader agent. The miner communicates its *bid* and the leader tells it if it is allocated the gold. Gold allocation triggers a plan to drop the gold to the depot. The subtle difference with our proposal is that in this case gold allocation is but a *signal*, so the miner is programmed to react to signals. The *causal* relation, that ties the plan to the event that activates it, is not expressed explicitly; it is in the structure of the protocol for interacting with the leader.

So, for instance, it is nothing that can be reasoned about nor it can be exploited for defining a programming methodology [2].

In our case, instead, the connection between the event "commitment detached" and the associated plan is not only causal, but the plan has the aim of satisfying the consequent condition of the commitment that triggers it (*drop(X, Y)*), i.e. of accomplishing an explicit and shared social engagement. The signal that notifies gold allocation is not relevant to the agent, at the point that it does not even appear in the code nor in the commitment. It is the detachment of the commitment itself that causes handling the gold. There is no need of knowing or using logics, that are internal to the protocol (artifact), for programming the agent. Social meanings are the key.

5 Conclusions

We presented JaCaMo+, an extension to JaCaMo that enables social behaviors into its agents. We started from the interaction protocols based on obligations proposed in [25]. However, obligation-based protocols reduce agent interactions to messages that an agent is obliged to send to another agent; that is, social relationships among agents are not handled directly. Thus, an obligation-based protocol can be adopted when an organization gives guidelines about how interactions should be carried on, but it is not applicable where similar guidelines are not available. To cope with these challenging situations, our intuition is to define an interaction in terms of goals and commitments. Commitments, in fact, are at the right level of abstraction for modeling directed relationships between agents. Moreover, since commitments have a normative power, they enable the agents to reason about the behavior of others.

One of the strongest points of JaCaMo+ is the *decoupling* between the design of the agents and the design of the interaction – that builds on the decoupling between computation and coordination done by coordination models like tuple spaces. The decoupling allows us to change the definition of the artifact without the need of changing the agents' implementation. So, in the gold miners scenario, allocation can be FIFO, based on the miners' position, or take into account further contextual information like day time, known differences in the equipment of the miners, difficulty in reaching the nugget location. All these different policies can be implemented in a way that des not have an impact on the miners' code.

JaCaMo (with interaction [25]) and JaCaMo+ do not equally support *autonomy.* JaCaMo with interaction just offers an agent to follow a predetermined path (a guideline) in which the agent has to fulfill a precise pattern of obligations. JaCaMo+, instead, offers an agent a tool, the interaction artifact, through which it can communicate with other agents and act together with others. The choice, however, of how and when being involved into an interaction remains up to the agents. The adoption of commitments, in fact, assures that an agent assumes the responsibility for a task only when, by its own choice, performs a specific action on the interaction artifact. An interaction that is based only on obligations hinders agents when they need to adapt to unforeseen conditions (flexibility) or when they need to react to unwanted situations (robustness). The agent, in fact, is not free to delegate obligations, schedule them differently, etc. All it can do is to perform the actions that, instructed by the interaction protocol, resolve its obligations. Protocols in [25] aim at defining guidelines to the use of resources in an organization. This, however, limits the modularity of *interaction* protocols because protocols depend on operations that are defined in the organization and there is no explicit association of which actions pertain to which roles. JaCaMo+ interaction protocols, instead, include the definitions of the needed operations, and specify which of them will empower the various role players.

The shift from obligations to commitments is beneficial in many respects. First, the autonomy of the agents is better supported because they are free in deciding how to fulfill their goals. It follows that agents are *deliberative*, and this paves the way to self-* applications, including the ability to autonomously take advantage from opportunities, and the ability of properly reacting to unexpected events (self-adaptation). Moreover, the interplay between goals and commitments opens the way to the integration of self-governance mechanisms into organizational contexts. Thus, our concluding claim is that directly addressing social relationships increases the robustness of the whole MAS.

In the future, we intend to investigate how agents can leverage on their deliberative capabilities, and use it not only to program interactions, but to plan social interactions. Moreover, the modular nature of the implementation facilitates the development of extensions for tackling richer, data-aware contexts [9,17,10]. We are also interested in tackling, in the implementation, a more sophisticate notion of social context and of enactment of a protocol in a social context [3], as well as to introduce a typing system along the line of [1].

Acknowledgements. This work was partially supported by the *Accountable Trustworthy Organizations and Systems (AThOS)* project, funded by Università degli Studi di Torino and Compagnia di San Paolo (CSP 2014).

References

1. Matteo Baldoni, Cristina Baroglio, and Federico Capuzzimati. Typing Multi-Agent Systems via Commitments. In *Post-Proc. of EMAS 2014, Revised Selected and Invited Papers*, number 8758 in LNAI, pages 388–405. Springer, 2014.
2. Matteo Baldoni, Cristina Baroglio, Federico Capuzzimati, and Roberto Micalizio. Empowering Agent Coordination with Social Engagement. In Proc. of *XIV Int. Conf. of the Italian Association for Artificial Intelligence*, 2015. To appear.
3. Matteo Baldoni, Cristina Baroglio, Amit K. Chopra, and Munindar P. Singh. Composing and Verifying Commitment-Based Multiagent Protocols. In *Proc. of 24th Int. Joint Conference on Artificial Intelligence, IJCAI 2015*, 2015.
4. Fabio L. Bellifemine, Giovanni Caire, and Dominic Greenwood. *Developing Multi-Agent Systems with JADE.* John Wiley & Sons, 2007.
5. Olivier Boissier, Rafael H. Bordini, Jomi F. Hübner, Alessandro Ricci, and Andrea Santi. Multi-agent oriented programming with JaCaMo. *Science of Computer Programming*, 78(6):747 – 761, 2013.

6. Olivier Boissier, Jomi F. Hübner, Alessandro Ricci, and Jaime S. Sichman. Multi-agent oriented programming, 2015. Tutorial at AAMAS 2015.
7. Rafael H. Bordini, Jomi Fred Hübner, and Daniel M. Tralamazza. Using *Jason* to implement a team of gold miners. In *CLIMA VII, Revised Selected and Invited Papers*, *LNCS* 4371, pages 304–313. Springer, 2006.
8. Rafael H. Bordini, Jomi Fred Hübner, and Michael Wooldridge. *Programming Multi-Agent Systems in AgentSpeak Using Jason*. John Wiley & Sons, 2007.
9. Federico Chesani, Paola Mello, Marco Montali, and Paolo Torroni. Representing and monitoring social commitments using the event calculus. *Autonomous Agents and Multi-Agent Systems*, 27(1):85–130, 2013.
10. Amit K. Chopra and Munindar P. Singh. Cupid: Commitments in relational algebra. In *Proc. of the 29th AAAI Conf*, pages 2052–2059. AAAI Press, 2015.
11. Natalia Criado, Estefania Argente, Pablo Noriega, and Vicent Botti. Reasoning about constitutive norms in BDI agents. *Logic Journal of IGPL*, 22(1):66–93, 2014.
12. Mehdi Dastani, Davide Grossi, John-Jules Ch. Meyer, and Nick A. M. Tinnemeier. Normative Multi-agent Programs and Their Logics. In *KRAMAS*, *LNCS* 5605, pages 16–31. Springer, 2008.
13. Yves Demazeau. From interactions to collective behaviour in agent-based systems. In *In: Proc. of the 1st. European Conf. on Cognitive Science. Saint-Malo*, 1995.
14. Frodi Hammer, Alireza Derakhshan, Yves Demazeau, and Henrik Hautop Lund. A multi-agent approach to social human behaviour in children's play. In *Proc. of the IEEE/WIC/ACM int. conf. on Intelligent Agent Tech.*. IEEE Comp. Soc., 2006.
15. Felipe Meneguzzi and Michael Luck. Norm-based behaviour modification in BDI agents. In *AAMAS (1)*, pages 177–184. IFAAMAS, 2009.
16. Felipe Meneguzzi, Pankaj R. Telang, and Munindar P. Singh. A first-order formalization of commitments and goals for planning. In *AAAI*. AAAI Press, 2013.
17. Marco Montali, Diego Calvanese, and Giuseppe De Giacomo. Verification of data-aware commitment-based multiagent system. In *Proc. of AAMAS*, pages 157–164. IFAAMAS/ACM, 2014.
18. Daniel Okouya, Nicoletta Fornara, and Marco Colombetti. An infrastructure for the design and development of open interaction systems. In *Post-Proc. of EMAS 2014, Revised Selected and Invited Papers*, number 8245 in LNAI, pages 215–234. Springer, 2013.
19. Andrea Omicini, Alessandro Ricci, and Mirko Viroli. Artifacts in the a&a meta-model for multi-agent systems. *JAAMAS*, 17(3):432–456, 2008.
20. Munindar P. Singh. An ontology for commitments in multiagent systems. *Artif. Intell. Law*, 7(1):97–113, 1999.
21. Munindar P. Singh. Commitments in multiagent systems some controversies, some prospects. In *The Goals of Cognition. Essays in Honor of Cristiano Castelfranchi*, chapter 31, pages 601–626. College Publications, London, 2011.
22. Pankaj R. Telang, Neil Yorke-Smith, and Munindar P. Singh. Relating Goal and Commitment Semantics. In *Proc. of ProMAS*, *LNCS* 7212. Springer, 2012.
23. Danny Weyns, Andrea Omicini, and James Odell. Environment as a first class abstraction in multiagent systems. *JAAMAS*, 14(1):5–30, 2007.
24. Pinar Yolum and Munindar P. Singh. Commitment Machines. In *Intelligent Agents VIII, 8th Int. WS, ATAL 2001*, *LNCS* 2333, pages 235–247. Springer, 2002.
25. Maicon R. Zatelli and Jomi F. Hübner. The Interaction as an Integration Component for the JaCaMo Platform. In *Post-Proc. of EMAS 2014, Revised Selected and Invited Papers*, number 8758 in LNAI, pages 431–450. Springer, 2014.