Endowing Business Artifacts with a Normative Coordination Layer

Matteo Baldoni, Cristina Baroglio, Federico Capuzzimati, Roberto Micalizio
Università degli Studi di Torino — Dipartimento di Informatica
c.so Svizzera 185, I-10149 Torino (Italy)
Email: {firstname.lastname}@unito.it

***Abstract*—We propose to enrich the artifact-centric approach in two ways. First, by relying on the Agent-Oriented Paradigm, the tasks acting on artifacts are organized in agents, seen as autonomous loci of control, whose execution is goal-driven. Second, the business artifact model is complemented by a normative dimension. Norms are used to represent the data lifecycle in a form that is inspectable and that can be reasoned upon by agents. Agents can therefore create expectations about the behaviors of others and hence, leveraging on the norms, agents can act on an artifact so as to entice, or oblige, others to act themselves. The paper discusses the advantages and consequences of this norm-aware enrichment, and outlines a possible realization based on social commitments.**

1. Introduction

The *artifact-centric* approach [5], [10], [8] is recently emerging as a viable solution for specifying and deploying business operations by combining both data and processes as first-class citizens. In particular, the notion of *Business Artifact*, initially proposed by Nigam and Caswell [12], opened the way to the development of a data-driven approach to the modeling of business operations. The data-driven approach counterposes a data-centric vision to the activity-centric vision, traditionally used when processes are explicitly modeled in terms of workflows. *Artifacts* are concrete, identifiable, self-describing chunks of information, the basic building blocks by which business models and operations are described. They are business-relevant objects that are created and evolve as they pass through business operations. A business artifact includes an *information model* of the data, and a *lifecycle model*, the latter capturing the key states through which data evolve together with state transitions. The lifecycle model is used both at runtime to track the evolution of artifacts, and at design time to distribute tasks. The presence of an explicit lifecycle gives business artifacts a semantics that differentiates them from other programming abstractions, like objects, active objects, and artifacts in the sense given to the concept by the A&A meta-model [13]. The lack of autonomy differentiates them from agents.

The work presented in this paper attacks a limit that business artifacts show: they do not provide the programmer with any means for designing and modularizing the *coordination* of those processes which should operate on them. It is a fact that business artifacts encapsulate data which are created and manipulated by many processes. For instance an order is created by a client who interacts with a seller, and is manipulated by the operations that make the transaction between the two proceed. The client, the seller, more in general the processes that operate on a business artifact need to agree on who should do what and sometimes even when to carry out a task of interest – e.g., payment and delivery must both occur otherwise the purchase transaction will not reach a happy end, and the two tasks are up to different parties in the transaction. So, on the one hand, a business artifact has a lifecycle that describes its evolution from creation to some state where it is considered as archivable. On the other hand, along the lines of activity theory – at the basis of A&A – a business artifact should also be a medium of coordination and interaction. However, this latter dimension is not supported by state-of-art literature on business artifacts. The proposal we present in this paper aims at overcoming the described lack of business artifacts with a multiagent systems (MAS) approach. Specifically, we claim that: (1) services by which it is possible to operate on business artifacts should be encapsulated and organized into *goal-oriented containers*; (2) it is necessary to introduce a *normative layer* to capture the behaviors that are expected of the parties. The paper motivates our research and our proposal, and illustrates it with the help of a simple purchase example.

2. Motivations

In order to understand how business artifacts are currently specified and used, we briefly introduce the BALSA methodology [6] as a significant representative of the current approaches to business artifacts. The BALSA methodology specifies a data-centric declarative model of business operations, and can be summarized in three steps: 1) identify the relevant business artifacts of the problem at hand and their lifecycles, 2) develop a detailed specification of the services (or tasks) that will cause the evolution of the artifact lifecycles, 3) define a number of ECA-rules (Event-Condition-Action) that create associations between services and artifacts. ECA-rules are the building blocks to define, in a declarative way, processes operating on data.

BALSA (and similar) is extremely interesting, in particular because it introduces a novel perspective on the

modeling of business processes. However, for what concerns coordination in presence of more business processes, the methodology simply refers to choreography languages and techniques proposed for service-oriented computing, despite the presence of the business artifacts which could themselves be natural instruments of coordination.

We deem the absence of an explicit use of the business artifacts to the aims of coordination as a significant flaw. In particular, in inherently *destructured* settings – like *cross-organizational settings*–, the involved actors are all peers, each of which has *its own* business goals, and acts in an autonomous way. Each actor does not know and does not care about the possible goals of others. Nevertheless, actors generally need to interact to achieve goals they would not be able to achieve alone. The interaction is a critical dimension that need to be explicitly modeled to coordinate the usage of shared resources. This poses the question of how to scale the business artifact model to *coordinate* autonomous entities. We see in the introduction of a *coordination model within business artifacts* the way to achieve this goal, and explain what we mean with a simple example.

Let us consider a purchase scenario, involving a merchant and a client. We claim that in order to coordinate the interaction between the two agents, it is necessary to add to the plain message exchange (which standard approaches to business processes envisage as the only means of interaction), one further abstraction that explicitly represents the engagements each player has towards the other. We also claim that business artifacts should trace such engagements and their evolution, in order to enable an effective agent coordination. For example, when offering to sell some goods, the merchant *commits to* the client to ship the items the client will pay for. Such a commitment is stored by the business artifact involved in the interaction between the two. Because of his *awareness* of such a commitment, the client, having paid for the goods, *expects* the shipment to occur. If this does not happen, the commitment progresses into a state of "violation" and this information, stored in the business artifact, provides a proof of the merchant's misbehavior. From a different perspective, a client is enticed to use a business artifact by the merchant's commitment, which makes explicit the course of interaction the merchant binds to, and creates a right on the client that such an expected course of action be respected (i.e., my payment will put an obligation on the merchant to ship the bought items or the merchant will violate the commitment). On the other hand, the merchant uses commitments inside the business artifact to entice interactions with potential clients – indeed, the obligation yielded by a commitment is activated only if a client pays for some goods.

In the example, the commitments that go along with a business artifact make explicit the behavior the agents are expected to stick to. They also have a normative flavour, as diverging behaviors will be considered as violations. This awareness causes agents to take part to an interaction only if they are fine with the commitments. As such, commitments provide a standard to define standards of interaction mediated by business artifacts. To realize this vision, we claim that: (1) services should be encapsulated and organized into *goal-oriented containers*; (2) it is necessary to introduce a *normative layer*. For what concerns (1), the Agent-Oriented Paradigm is a good candidate. In particular, the Agent and Artifact meta-model (A&A) [13] has already shown how artifacts can be used as environment components that mediate agents' interactions. However, artifacts in the A&A model are radically different from the business artifacts because they *do not come* with an explicit information model for data, and they do not exhibit data lifecycles. Thus, this information cannot be exploited at design time, nor at runtime, to reason about which actions an agent should take. Concerning (2), the normative layer would provide an explicit representation of the business artifacts lifecycles, and of how coordination is expected to occur. Such a representation would allow agents to reason about the use of business artifacts and to create *mutual engagements* for driving their activities. Indeed, we envisage engagements as encoding *causal relations* between the actions of an agent and the goals and actions of another, with a normative power that would allow each agent to have expectations on the behavior of the others. In the purchase example, it is easy to see how the introduction of a norm in form of the commitment *whenever a customer pays, the merchant will ship the goods*, would enhance coordination. The customer now knows that after service *pay*, the merchant will be pushed to consider the service *ship-goods* as one of its next goals, otherwise it will violate the norm and will be sanctioned for that. This provides the customer a guarantee about the achievement of its own goal (or to recoup its losses). An explicit normative layer plays a central role both at the design time, to verify whether all the engagements can converge towards their satisfaction, and at running time to monitor the execution of a system and determine the violation of engagements. In this paper we introduce the notion of normative business artifacts as a means to extend the artifact-centric approach with a normative layer, where engagements and norms are expressed in terms of social commitments [14]. The introduction of a normative layer in the more general setting of business processes is seen as desirable also in [16].

3. Coordination via Normative Business Artifacts

Business artifacts are, by definition, data-aware: they consider data as a first-class primitive that drives the construction of process models [5]. Artifacts, however, are not an end in themselves: they are business relevant entities that are created, accessed, and manipulated by different services along a business process. We now show how to introduce a normative layer so that business artifacts support coordination.

Destructured business processes call for a modularization of the control flow. Agent-oriented programming [7], [19] is conceived exactly for handling multiple and concurrent control flows. Two elements are central in agent-oriented programming: the *agents* and the *environment*.

Agents, as abstractions of processes, possess their own control flow, summarized as the cyclic process in which an agent observes the environment (updating its beliefs), deliberates which intentions to achieve, plans how to achieve them, and finally executes the plan [7]. Beliefs concern the environment. Intentions lead to action [19], meaning that if an agent has an intention, then the expectation is that it will make a reasonable attempt to achieve it. In this sense, intentions play a central role in the selection and the execution of actions, which represent the innate capabilities agents have to modify their environment. Among others, (business) artifacts (see A&A-meta model [13]) are privileged elements of an environment. In particular, in contexts where agents cannot achieve their goals on their own, but need to interact with other agents to do so, artifacts provide shared resources that agents will use to mediate their interactions.

We claim that business artifacts should be *norm-aware* in two ways. First, the *lifecycle* of a business artifact should be made explicit by way of norms that specify how data evolve. The agents (i.e., the artifact users), will be able to inspect and reason upon them to decide if and how to operate on an artifact to obtain some result. Second, agents need to coordinate and regulate their interaction *while using* the business artifacts to achieve their goals. Given these two bodies of norms, agents will apply reasoning techniques to plan proper *coordination* that, possibly without violating any norm, will lead to goal achievement. This is possible because norms enable the creation of *expectations* and *commitments* among agents. Moreover, given an explicit representation of such elements it will be possible to realize systems of *accountability* to discourage or to detect and explain deviant behavior [4].

Even though data-awareness and norm-awareness are by and large orthogonal to BDI [19] notions, it is natural to think of agents as BDI agents for a seamless integration of all the aspects of deliberation, including the awareness of data and of their lifecycles. For instance, an agent, that is involved in handling orders, may conclude that, since it has to pick up three items in the warehouse, since each such item is to be packed, since all packagings are performed by a same other agent, and since one of its goals is saving energy, it is preferable to pick them up altogether, and deliver them to the other agent only afterwards, instead of picking and delivering one item at a time. Data-awareness here is awareness that three items of a same kind are requested. Norm-awareness that items are picked because each of them is part of some order, whose lifecycle says that after being picked they will be packed. Again data-awareness allows our agent to know that all parcels are to be made by a same other agent.

Relying on agent-oriented programming is promising also because a the agent-based model allows to naturally tackle the issue of coordination by introducing the concept of *norm* [18]. The deliberative cycle of agents is affected by the norms and by the obligations these norms generate as a consequence of the agents' actions. The limit of current agent-based approaches is that they provide no holistic proposal where constitutive norms are used also to specify data operations, and where regulative norms are used to create expectations on the overall evolution of the system (agents behavior and environment evolution).

3.1. Environment/Information systems based on normative business artifacts

Figure 1 describes the high-level architecture of the kind of system we imagine: (1) involving business artifacts and agents (with their goals), and (2) holistically norm-aware. Agents interact with each other and with the environment by creating and modifying data which belong to an information system and that are reified by business artifacts. They are goal-driven and capable of coordinating with other agents by creating and exploiting commitments, obligations, permissions, and prohibitions. The conceptual model of the information system is described in terms of the norms that regulate the evolution of data, that is, data lifecycles, capturing how data pass from one state to another as a consequence of actions that are performed by some agent. Moreover, business artifacts will include all those normative elements that regulate the coordination of the agents that interact by way of the artifact. All this information is available to the interacting agents in a form that allows agents to reason on it. The agents are aware of the current state (of the lifecycle) of the data, as well as of the obligations, prohibitions, commitments, permissions put on them, and thus they are aware of the tasks expected of them and of their parties. At any time it is possible to check the execution, identifying pending tasks and who is responsible of them, as well as possible violations (e.g. of obligations or commitments), which may activate procedures specifically designed to handle the case.

4. Building Normative Business Artifacts in JaCaMo+

In this section we explain how the normative business artifact we propose can be implemented by relying on the 2COMM/JaCaMo+ framework [3]. We refer to an implementation where the BDI agents are implemented in the Jason agent programming language, and where agents share artifacts, whose creation and manipulation involves an explicit creation and manipulation of *social commitments* [14]. Social commitments provide the normative layer and enable the coordination of the goal-driven agents.

We exemplify the implementation in the purchase scenario. In this scenario each agent has its own goals: the merchant has the goal of selling goods, while the customer has the goal of getting some goods. We show how they can achieve their goals by using a business artifact as the only means of coordination. To this end we need to present both sides of the interaction: the normative business artifact, on the one side, and how the agents use it, on the other side.

Let us consider first the business artifact. Figure 2 shows the business artifact representing the transaction occurring between a merchant and a customer. This artifact follows the

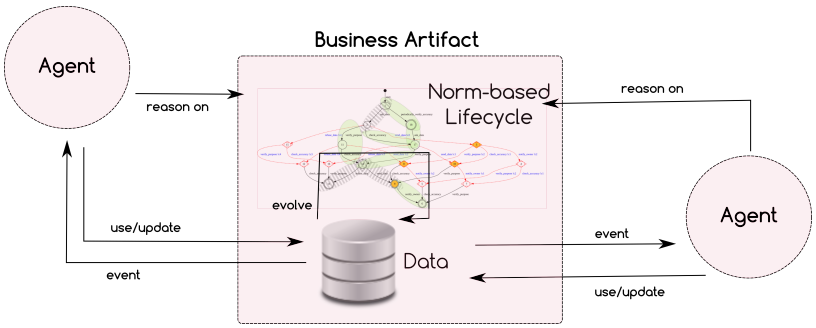

Figure 1. Environment/Information system based on normative business artifacts.

principles of the business artifact proposed in the BALSA meta-model: a data model is defined to trace relevant pieces of information; namely, the merchant and customer identifiers, the item that can be sold by the merchant and the maximum number of pieces that are available. While this information is provided at the time the business artifact is created, three further pieces of information (namely, quotation, quantity and order) are, instead, the result of the operations above the artifact performed by the agents using it. These operations are captured by the business artifact lifecyle showed here as an automaton: the customer asks the merchant for a quotation of a given quantity of items it wants to buy. Once the quotation is provided, the customer either decides to reject or accept the quotation. In the first case, the business artifact achieves a final state and can be archived. In the second case, a new order number is created to trace the shipping and the payment of the goods. Note that no ordering between shipping and payment is imposed. After the payment, the merchant issues the payment receipt, and the business artifact can be archived.

As noted above, such a business artifact is not sufficiently rich to trace the causal relationships. The customer may have an expectation about the behavior of the merchant, raised by its experience in purchasing things, that the payment of an item will be followed by the merchant giving the item but this expectation has no normative power. We, therefore, extend this artifact with a normative layer that, as anticipated, is expressed in terms of a set of commitments. A social commitment $C(x, y, s, u)$ captures that agent x (debtor) commits to agent y (creditor) to bring about the consequent condition u when the antecedent condition s holds (s and u are conjunctions or disjunctions of events). Only the debtor of a commitment can create it. When s is $true$ the commitment is *detached* and turns into an obligation on the debtor. When u is $true$ the commitment is *satisfied*. A detached commitment that is canceled or whose consequent becomes $false$ is *violated*.

To realize a normative business artifact, thus, it is sufficient to associate each operation on the business artifact with operations (e.g. create, discharge, etc.) on one (or more) commitment(s). It follows that a normative business artifact, besides representing the chunk of information at hand, maintains also the created commitments, that can be inspected by the agents. Specifically agents will be notified of the changes to the business artifact state which include changes occurred to the commitments. Among other events, they will be aware of the detachment of commitments of which they are debtors, and of the satisfaction (violation) of commitments of which they are creditor.

```
commitment ShipGoods merchant to customer
  create quote(quantity, customer)
  detach accept(quotation, quantity, customer)
  discharge ship(customer)
  release reject(price, quantity, customer)

commitment EmitReceipt merchant to customer
  create quote(quantity, customer)
  detach paid(customer)
  discharge emit(customer)
  release reject(price, quantity, customer)

commitment PayForGoods customer to merchant
  create accept(quotation, quantity, customer)
  detach ship(customer)
  discharge paid(customer)
```

Listing 1. The normative layer in the purchase scenario

In our example, the normative layer is given by the set of commitments in Listing 1. For the sake of readability, the commitments are expressed following the syntax of the Cupid language [9]. The first commitment, *ShipGoods*, is created by the merchant when it executes the *quote* operation. That is, besides giving a value to the *quotation* information, the *quote* operation also commits the merchant towards the customer. Such a commitment is discharged when the customer accept the quotation, and discharged when the merchant ships the bought goods to the customer. Also the second commitment, *EmitReceipt*, is created by the merchant by the execution of the *quote* operation. In this case, the commitment is detached when the customer pays for the goods, and it is discharged when the merchant emits the receipt towards the customer. Both these two commitments are released by the customer when it rejects the quotation provided by the merchant. The last commitment, *PayForGoods*, is created by the customer when it accepts the merchant's quotation. The commitment is detached when the merchant has shipped the goods, and discharged when the customer pays for them.

Now, let us briefly review the resulting *normative* business artifact. As before, the merchant advertises, by creating the artifact, some item to be sold, and specifies the number

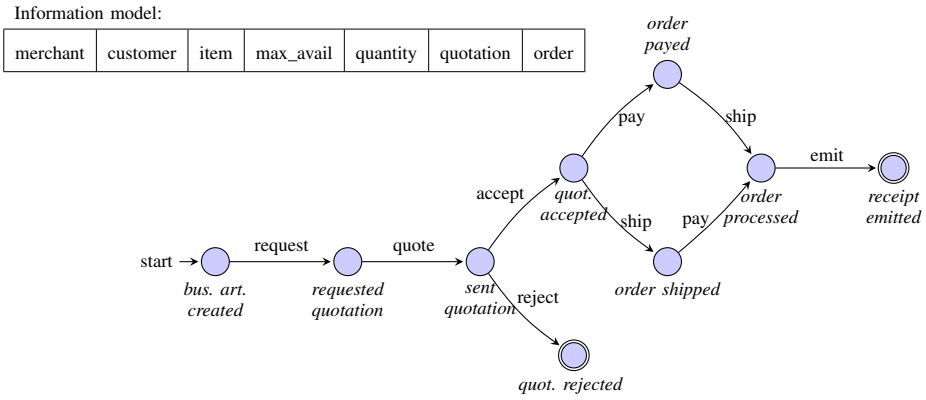

Figure 2. The business artifact for the purchase scenario.

of available units. An interested customer, by inspecting the artifact, can now see the commitments that the merchant is willing to take towards the customer. That is, the customer can create expectations about the merchant's behavior that have a *normative power*. The customer is, thus, enticed to accept the quotation because the presence of the commitment, as part of the information provided by the business artifact, yields that this action will create an obligation on the merchant to deliver the goods that will make it achieve his goal. On the other hand, the customer will see also the commitments it will take in favor of the merchant, should it join the interaction. So, if the customer starts an interaction by requesting a quotation for a number of units, the merchant will provide such a quotation and, at the same time, it will create a commitment to: 1) ship the goods, provided that the customer accepts the quotation (*ShipGoods*), and 2) emit a receipt upon the payment for the goods (*EmitReceipt*).

Note how the operations performed by agents on the business artifact make the commitments progress. For instance, the customer's acceptance of the quotation has several effects: 1) on the data side, a new order number is created to trace shipping and payment; 2) the customer commits towards the merchant to pay for the goods once they are delivered (*PayForGoods*); 3) commitment *ShipGoods* is detached, and then the merchant is now asked to ship the goods. As a final comment about the artifact, note how the commitments do not impose any ordering about the payment and shipping. In fact, the customer could pay as soon as it accepts the quotation, assured by the existence of commitment *ShipGoods* that pushes the merchant to actually ship the goods.

A natural way to implement normative business artifacts is to rely on the 2COMM framework [3]. In 2COMM normative business artifacts are reified as commitment-based protocol artifacts, that are provided by the framework as Java classes. Business artifact operations are mapped into protocol actions, whereas the data dimension is captured by the notional social state that protocol artifacts maintain. Indeed, the social state kept by a protocol artifact traces both the data required for the interaction, and the commitments, together with their states, that are created and manipulated along the interaction.

Let us now discuss the other side of the interaction, that is, how the agents use a normative business artifact. To exemplify the agents, we use here the JaCaMo+ framework, consisting in the well-known JaCaMo multi-agent platform enriched with commitment protocols provided by means of the 2COMM framework. Below, an excerpt of the merchant agent program. In this first plan, the merchant is solicited to act by the reception of a *requestedQuotation* event, that comes from a customer through the business artifact. The body of the plan consists in the execution of *quote*, which sends a quotation to the customer and causes the creation of the merchant's commitments *ShipGoods* and *EmitReceipt*. The second plan captures the detachment of the *ShipGoods* commitment. The detach of the commitment is indeed an event generated by the artifact the merchant is focusing on, and it is the consequence of the accept operation performed by the customer. The body of the plan consists in the *ship* operation. Finally, the third plan captures the detachment of the *EmitReceipt* commitment, also in this case the body of the plan aims at discharging the commitment.

This example shows how the two agents, merchant and customer, can interact with each other by using a business artifact as an interaction medium. The normative layer assures that each agent will be willing to discharge its own commitments to avoid their violation. Notably, each agent achieves the goal it had at the beginning of the interaction; namely, the merchant gains money by selling goods, and the customer has the goods it is interested in. These two goals, however, are not immediately apparent when one considers the business artifact depicted in Figure 2. The two final states, in fact, just denote situations in which the business artifact can be archived, but do not specify any goal achievement by the involved agents. One further advantage of adding the normative layer is the possibility of making explicit some of the goals and services agents make available to others. This piece of knowledge can be used by the agents in a practical reasoning step (see [17]) to decide whether to

```
+requestedQuotation(Quantity, Customer_Id)
    <- quote(UnitPrice*Quantity, Quantity, Customer_Id).
+cc(My_Role_Id, Customer_Role_Id, accept(Price, Quantity, Customer_Role_Id),
    ship(Customer_Role_Id),"DETACHED") : enactment_id(My_Role_Id)
    <- ship(Customer_Role_Id, Quantity).
+cc(My_Role_Id, Customer_Role_Id, paid(Customer_Role_Id), emitReceipt(Customer_Role_Id),"DETACHED")
    : enactment_id(My_Role_Id)
    <- emitReceipt(Customer_Role_Id).
```

join the artifact.

5. Conclusions

The presented work is strictly related to the problem of interaction in multiagent systems. In these systems, interaction is mainly focused on the modeling of communication patterns (*protocols*), which are concerned with the sequence of messages that can be exchanged between two communicating agents, but disregard the information conveyed by these messages. Recent approaches such as HAPN [20] and BSPL [15] have started to consider also the information dimension. HAPN is formally based on automata where nodes represent states of the interaction and transitions between nodes represent the messages that can be exchanged. Transitions have a complex structure since for each message it is possible to define a guard condition on message sending. A similar approach is BSPL where the information flow is decomposed in a number of "simple protocols", each defining the schema of the messages that can be exchanged together with their parameters. Parameter are decorated as *in* or *out* (meaning it is received or emitted). BSPL provides a formal framework in which it is possible to verify properties such as liveliness and safety of a protocol. Both HAPN and BSPL, however, show some weaknesses in properly handling information. In HAPN, for instance, guards, that enable message sending, may refer to information which is not carried by the message itself, but rather maintained in an external information system, which is not an integral part of the HAPN proposal, and hence the complete verification of an interaction is not actually achievable. BSPL, on the other hand, assumes a distributed view of information. Each participant has its own knowledge base, and the progression of the interaction makes the local knowledge bases evolve. The problem, in this case, is that each participant has just a local view of the information lifecycle. Thus, an agent cannot create expectations about the behaviors of other participants as a consequence of the messages it sends. The approach we propose overcomes these limitations. Business artifacts abstract an information system, and provide the environment in which the agents, which are autonomous loci of control, interact. Both business artifacts and agents are first-class components. The autonomy and flexibility of the agents are preserved and supported; moreover, it is possible to reason both on the evolution of the business artifacts and on the interaction. This work can be extended along three main lines of research. First of all, an explicit normative layer paves the way to formal verification techniques for cross-organizational business processes. In this respect, the notion of *accountability* is rapidly gaining importance since, when more organizations come into play, it is even more important to trace back who is responsible for what. First steps can be found in [4]. Another promising extension is to understand how agents could plan the use of business artifacts for reaching their goals. An initial attempt to use social commitments in planning has been discussed in [2], but business artifacts are yet to be considered. Finally, the standardized lifecycle of commitments can be the key for developing an agent programming methodology, similar to the one discussed in [1]. The idea is to program agents so that they can properly tackle part of the events that are generated in the business artifacts of their interest; specifically, the state transitions that occur to commitments in which they are involved. To conclude, we mention RAW-SYS [11], which enriches the prescriptive process model with data-awareness. Although RAW-SYS looks similar to a (normative) business artifact, the objectives of the two models are quite different. RAW-SYS is essentially a framework for verifying business processes taking into account both the control- and the data-flows. A normative business artifact, instead, aims at coordinating autonomous agents.

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, Federico Capuzzimati, and Roberto Micalizio. Empowering agent coordination with social engagement. In *AI*IA 2015, Advances in Artificial Intelligence, LNCS 9336*, pages 89–101, 2015.

[2] Matteo Baldoni, Cristina Baroglio, Federico Capuzzimati, and Roberto Micalizio. Exploiting social commitments in programming agent interaction. In *PRIMA 2015: Principles and Practice of Multi-Agent Systems*, pages 566–574, 2015.

[3] Matteo Baldoni, Cristina Baroglio, Federico Capuzzimati, and Roberto Micalizio. Commitment-based agent interaction in JaCaMo+. *Fundamenta Informaticae (to appear)*, 2017.

[4] Matteo Baldoni, Cristina Baroglio, Katherine M. May, Roberto Micalizio, and Stefano Tedeschi. Computational accountability. In *Proceedings of the AI*IA Workshop on Deep Understanding and Reasoning*, pages 56–62, 2016.

[5] K. Bhattacharya, N. S. Caswell, S. Kumaran, A. Nigam, and F. Y. Wu. Artifact-centered operational modeling: Lessons from customer engagements. *IBM Syst. J.*, 46(4):703–721, 2007.

[6] K. Bhattacharya, R. Hull, and J. Su. *A data-centric design methodology for businessprocesses*, pages 503–531. Handbook of Research on BusinessProcess Modeling. IGI Publishing, 2009.

[7] Michael E. Bratman. What is intention? In P. Cohen, J. Morgan, and M. Pollack, editors, *Intensions in Communication*, pages 15–31. MIT Press, Cambridge, MA, 1990.

[8] Diego Calvanese, Giuseppe De Giacomo, and Marco Montali. Foundations of data-aware process analysis: a database theory perspective. In *Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS*, pages 1–12. ACM, 2013.

[9] Amit K. Chopra and Munindar P. Singh. Cupid: Commitments in relational algebra. In Blai Bonet and Sven Koenig, editors, *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 2052–2059. AAAI Press, 2015.

[10] David Cohn and Hull Richard. Business Artifacts: A Data-centric Approach to Modeling Business Operations and Processes. *IEEE Data Eng. Bull.*, 32(3):3–9, 2009.

[11] Riccardo De Masellis, Chiara Di Francescomarino, Chiara Ghidini, Marco Montali, and Sergio Tessaris. Add data into business process verification: Bridging the gap between theory and practice. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 1091–1099, 2017.

[12] A. Nigam and N.S. Caswell. Business artifacts: An approach to operational specification. *IBM Systems Journal*, 42(3):428 – 445, 2003.

[13] Andrea Omicini, Alessandro Ricci, and Mirko Viroli. Artifacts in the A&A meta-model for multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 17(3):432–456, December 2008. Special Issue on Foundations, Advanced Topics and Industrial Perspectives of Multi-Agent Systems.

[14] Munindar P. Singh. An ontology for commitments in multiagent systems. *Artif. Intell. Law*, 7(1):97–113, 1999.

[15] Munindar P. Singh. Information-driven interaction-oriented programming: BSPL, the blindingly simple protocol language. In *10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 491–498, 2011.

[16] Munindar P. Singh. NoBPM: Supporting Interaction-Oriented Automation via Normative Specifications of Processes, 2015. Invited talk, BPM.

[17] Pankaj R. Telang, Munindar P. Singh, and Neil Yorke-Smith. Relating Goal and Commitment Semantics. In *Post-proc. of ProMAS*, volume 7217 of *LNCS*. Springer, 2011.

[18] Göran Therborn. Back to norms! on the scope and dynamics of norms and normative action. *Current Sociology*, 50:863–880, 2002.

[19] Michael J. Wooldridge. *Introduction to multiagent systems, 2nd edition*. Wiley, 2009.

[20] Nitin Yadav, Lin Padgham, and Michael Winikoff. A tool for defining agent protocols in HAPN: (demonstration). In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS*, pages 1935–1936, 2015.