Revisiting the DEMO Transaction Pattern with the Unified Foundational Ontology (UFO)

Tanja Poletaeva[1], Giancarlo Guizzardi[2,3], João Paulo A. Almeida[3], Habib Abdulrab[1]

[1] LITIS lab., INSA de Rouen, Rouen, France
ta.poletaeva@gmail.com, habib.abdulrab@insa-rouen.fr
[2] Free University of Bozen-Bolzano, Italy
[3] Ontology and Conceptual Modeling Research Group (NEMO), Federal University of Espírito Santo, Vitória, ES, Brazil
{gguizzardi, jpalmeida}@inf.ufes.br

Abstract. In this paper, we revisit the DEMO transaction pattern in light of the domain-independent system of categories put forth by the Unified Foundational Ontology (UFO). In this process, we treat social relationships in the scope of the DEMO transactions as objectified social entities, and thereby separate the behavioural and structural aspects of the transaction pattern and clarify their interplay. Further, we represent the pattern in the OntoUML ontology-driven conceptual modeling language. The revisited pattern can be embedded in broader enterprise ontologies and reference conceptual models based in UML. The proposed OntoUML models can also be further refined to account for and consider different organizational implementations of business transactions. We demonstrate the proposed representation by applying it to OMGs EU-Rent case.

Keywords: Foundational ontology, enterprise ontology, DEMO transaction pattern, organizational implementation.

1 Introduction

Since 1960s, conceptual modeling is widely adopted for knowledge communication among human users [1]. The importance of enterprise conceptual modeling in enterprise engineering and transformation [2] has encouraged the development of various enterprise modeling methods. Nowadays, there is a growing interest in approaches that employ ontologies as theoretical tools for improving conceptual models. Among such approaches, there is a mature DEMO methodology (the Design and Engineering Methodology for Organizations) [3], which comprises the DEMO enterprise ontology, the ontology-based enterprise modeling language, and the modeling method.

Despite the conceptual quality of DEMO, we observe that there are still opportunities for clarification and generalization of its conceptual basis, in particular considering some aspects of social relationships that evolve in business transactions. In addition to that, there are little guidelines on how to integrate knowledge conceptualized with DEMO to other (non-DEMO based) organizational conceptual models that are widely employed in practice (such as, e.g., reference organizational models captured

in UML). These organizational conceptual models can play an important complementary role in the DEMO methodology, when used to represent the types of objects involved in business interactions and their properties in addition to elements of the DEMO conceptual models. Moreover, a broader ontological account of the DEMO models is required for modeling of organizational implementations in compliance with the core enterprise knowledge. We believe that coherent conceptual models of an organization at different levels of details support understanding and communication of desired organizational transformations.

We address these aforementioned issues in this paper. Firstly, we revisit a central notion of the DEMO enterprise ontology, namely, ï the transaction pattern. The transaction pattern is a uniform communication pattern, which was proposed by Dietz for modeling of business interactions [3]. We revisit the nature of social relationships in the scope of the DEMO transaction pattern based on the domain-independent system of categories put forth by the Unified Foundational Ontology (UFO) [4]. In this process, we treat these social relationships as objectified social entities, and thereby separate the behavioural and structural aspects of the transaction pattern and clarify their interplay. Secondly, we represent the transaction pattern using the UFO-based OntoUML ontology-driven conceptual modeling language [4]. The revisited pattern can be embedded in broader enterprise ontologies and reference conceptual models based in UML. The proposed OntoUML models can also be further refined to account for and consider different organizational implementations of business transactions. We demonstrate the proposed representation by applying it to OMGs EU-Rent case that was the subject of DEMO analysis in [5].

OntoUML is an example of a conceptual modeling language whose meta-model has been designed to comply with the ontological distinctions and axiomatization of UFO [4, 6]. OntoUML has been successfully employed in a number of industrial projects in several different domains, such as petroleum and gas, digital journalism, complex digital media management, off-shore software engineering, telecommunications, retail product recommendation, and government [6]. A recent study shows that UFO is the second-most used foundational ontology in conceptual modeling and the one with the fastest adoption rate [7]. Moreover, the study also shows OntoUML is among the most used languages in ontology-driven conceptual modeling (together with UML, (E)ER, OWL and BPMN).

The outline of this article is as follows. In Sect. 2, we summarize the original DEMO ontological commitments related to the transaction pattern. In Sect. 3, we briefly explain some of the ontological foundations employed to revisit the pattern, including some key social notions in UFO, along with the fragment of OntoUML adopted here. In Sect. 4, we use the UFO notions to reconceptualize the DEMO transaction pattern with transactions analyzed from two complementary perspectives: a structural one, in which transactions are considered objectified social relations, and a behavioural one, in which transactions are considered occurrences or events. The section also presents OntoUML models of the transaction pattern, showing the interplay between two perspectives. Sect. 5 illustrates the application of the proposed OntoUML representation for modeling of organizational implementation variables, extending the models of Sect. 4. Finally, Sect. 6 presents our conclusions.

2 The DEMO Transaction Pattern

In the DEMO Enterprise Ontology [3], Dietz claimed to have proposed what he terms the ñmolecular structureò of business interactions. In his view, a business transaction is a minimal social conversation carried out between two social individuals, one of which (the *initiator*[1]) initiated the conversation in order to delegate an achievement of his/her goal to another party. The party who accepted the request for achievement of someoneôs goal is called the *executor*. In line with Habermas [8], Dietz relies on a desire of both parties to reach a consensus in a business deal.

Hereafter, we summarize the ontological commitments of the DEMO Enterprise Ontology [3, 9] related to the transaction pattern.

- C1. By performing *coordination acts* (*C-acts*), social individuals of an enterprise enter into and comply with social commitments towards each other regarding the product to be brought about.
- C2. The original new thing that is created by a C-act is a commitment (also named a *coordination fact*).
- C3. Two time aspects of coordination facts are distinguished: the event time and the settlement time.
- C4. By performing *production acts* (*P-acts*), social individuals in an organization create products.
- C5. There is a one-to-one relationship between *transaction kinds* and product kinds. Transaction kind is a basic property of every *transaction*.
- C6. *Actor role* is the authority to fill the *executor role* in transactions of a particular transaction kind. It includes (by definition) the authority to be the initiator in transactions of a number of (other) transaction kinds.
- C7. An *actor* is a social individual (subject) in the quality of filling an actor role.
- C8. A transaction involves two actors, one as the *initiator* and one as the *executor*.
- C9. The *process of a transaction* is a temporally ordered sequence of coordination acts of the initiator and the executor, starting from a requesting coordination act of the initiator.
- C10. The process of a transaction is a path, possibly including iterations, through a *universal transaction pattern.*
- C11. A complete transaction goes off in three consecutive phases: the *order phase*, the *execution phase*, and the *result phase*. The process of a transaction in the order and the result phases is a sequence of coordination acts. In the execution phase, the executor performs some production act(s).

3 UFO and OntoUML

In this section, we present a subset of OntoUML language that is employed here for the representation of the DEMO transaction pattern. We also briefly discuss the UFO concepts underlying the OntoUML constructs used. Finally, we summarize the UFO

[1] In this section, we introduce the terms from the DEMO vocabulary in *italics*.

ontological commitments about social entities [10] that are relevant to the DEMO commitments discussed in Sect. 2.

3.1 OntoUML

OntoUML is an ontologically well-founded version of UML (Unified Modeling Language) whose metamodel reflects a number of ontological distinctions and axioms put forth by UFO [4, 6]. This means that an OntoUML representation of state of affairs in reality is unambiguously interpreted based on domain-independent ontological categories.

In OntoUML, class constructs stereotyped by «Kind» represent object types that supply a uniform *principle of identity*[2] for their instances. Specializations of classes representing kinds are stereotyped as «SubKind», «Role», or «Phase». All these specializations inherit their principle of identity from «Kind» types. While object types stereotyped by «Kind» and «SubKind» are necessarily applied to their instances in every possible world (i.e., these types are *rigid*), instances of «Role» and «Phase» types can cease to be instances of these types without ceasing to exist and without altering their identity. Moreover, while instances of «Phase» types are characterized by a change of their intrinsic property(s), instances of «Role» types are characterized by a relational property(s) acquired in relationships with other entities.

«Category» and «RoleMixin» types represent an abstraction of properties that are common to multiple «Kind» types and, therefore, do not carry a unique principle of identity for their instances. Properties associated with «Category» types are rigid and relationally-independent, while properties associated with «RoleMixin» types necessarily represent an abstraction of contingent (or *anti-rigid*) properties that are common to different «Role» types.

In addition to the aforementioned object types, OntoUML class elements represent types of existentially dependent individuals that can only exist by inhering in other individuals. Such individuals are called *moments*. Those moments that inhere in one single individual are categorized as «Mode» or «Quality» types. While qualities (also called *individual qualities*) are moments that change in a particular space of possible values (e.g. a color, a temperature, a weight), modes are complex individual moments that can have their own qualities that take their respective values in multiple independent value dimensions (e.g., a symptom, a capacity, a complex intention). While inhering in a single individual, some modes and qualities can *externally depend on* (possibly a multitude of) other individuals that are independent from their bearers. Moments that existentially depend on two or more individuals are categorized as «Relator» types.

Instances of «Event» types are *perdurants*. Perdurants unfold in time accumulating temporal parts. They are defined by the sum of their parts (their constituent subevents) and they bear to each other a number of temporal ordering and causality relations. Moreover, perdurants are *manifestations* of dispositional properties of moments

[2] The terms from the UFO vocabulary, which are introduced in addition to OntoUML stereotypes, are highlighted in *italics*.

(qualities, modes, and relators). Finally, perdurants are immutable in all their parts and all their properties [11].

Moments are connected to their bearers via existential dependence relations. In OntoUML, intrinsic moment types (quality and mode types) are connected to the type representing their bearers via a relation of «characterization». This relation is mapped at the instance level to a relation of *inherence*, i.e., a particular type of functional external dependence relation; relator types, in contrast, are connected to the type of entities they relate (bind, mediate) via an association stereotyped as «mediation», which is mapped at the instance level to a non-functional type of existential dependence relation.

3.2 UFO-C: the Social Layer of UFO

In a social context, the UFO-C part [10] of UFO distinguishes between agentive and non-agentive substantial individuals. Agentive individuals (or *agents*) are capable of bearing special kind of moments named *intentional moments*. Intentional moments can be further specialized into *mental moments* (including *beliefs*, *desires* and *intentions*) and *social moments*. Moreover, each type of intentional moments necessarily has a propositional content, which may be matched by certain situations in reality. Thus, the intentionality of agents should be understood as the capacity of their properties to refer to possible situations of reality.

Among other types of intentional moments, *Intentions* refer to desired state of affairs to which an agent *internally commits* at pursuing. For this reason, intentions cause the agent to perform *actions*. Actions are intentional events, i.e., events with the specific purpose of satisfying the propositional content of some intention of an agent. The propositional content of an intention is termed a *goal*. UFO contemplates a relation between situations and goals such that a situation may satisfy a goal.

Communicative acts (special kinds of actions) can create *social moments* (*commitments* and *claims*) inhering in the agents involved in these communicative acts. In opposite to internal intentional moments, social commitments and claims *inhere* in one agent and are *existentially depend* on another. If a social commitment inheres in an individual X and is externally dependent on another agent Y (i.e., it is a commitment of X towards Y) then there is a dual *social claim* inhering in Y and which is externally dependent of X (i.e., the claim of Y towards X). In other words, commitments and claims always form a pair that shares a unique propositional content [10]. Two or more pairs of mutually dependent commitments and claims form a kind of social relationship between involved social individuals. This social relationship is termed in UFO-C a *social relator*. Social commitments and claims are often associated with internal commitments (self-commitments).

4 Ontological Analysis and Representation of the Transaction Pattern

In this section, we propose to align the original ontological commitments for the DEMO transaction pattern given in Sect. 2 employing the conceptual notions put forth by UFO. In this process, we elaborate on the benefits of considering business transactions as endurants (more specifically as social relators) together with reifying corresponding transaction events as the context of business interactions. In addition to revisiting the conceptual aspects of the transaction pattern in light of the UFO, we provide a representation of the revisited pattern using OntoUML models which can be later used as a basis for extension in order to model enterprise-specific settings. In this section as well as in Sect. 5, we write the elements of the proposed models in *italics*. Stereotypes and the names of relations start with lowercase, while types are capitalized.

4.1 Transactions as Endurants

A central notion in the transaction pattern is the notion of transaction. We propose a transaction should be understood as a relator, composed of social commitments and claims made by involved actors in their negotiation about an achievement of some shared goal (i.e., a production result), as well as by other relational qualities acquired by actors in this negotiation. Thus, a transaction can be represented by a relator mediating two actors, which play the roles of *Initiator* and *Executor* (Fig. 1). A particular role played by an agent in a transaction is defined by the type of his commitments and claims. An actor is the executor, when he commits himself to a requesting actor (the initiator) to achieve a production result. Although in this paper we excluded the self-activating actors from consideration, this additional constraint cannot be expressed in UML.

Actor specializes the notion of UFO Agent; an *Actor* is an Agent that participates (contingently) in a *Transaction*, which suggests that the *Actor* type should be stereotyped «role». We have opted to represent the type *Actor* as a *role mixin* (instead of *role*) in order to cater for the possibility that they obey different principles of identity (for example to allow for individuals and for teams to be considered *Actors*).

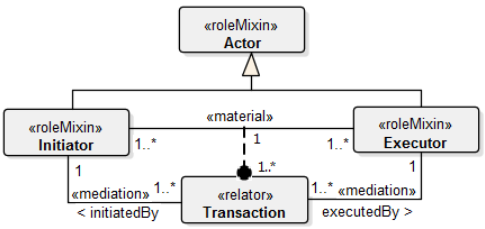


Fig. 1. Participation in a transaction

By applying the powertype pattern from a Multi-Level Theory [12] to C6, C7, and C8 commitments from Sect. 2, we propose modeling of the initiator and the executor of a transaction as instances of the *Actor Role* powertype (Fig. 2), i.e. a rigid sortal whose instances are types. Following [13], we extended the OntoUML metamodel by introducing the stereotype «hou» to represent high-order universals. All roles specializing *Initiator* are instances of the *Initiator Actor Role* powertype, while all roles specializing *Executor* are instances of the *Executor Actor Role* powertype. The generalization set of *Initiator Actor Role* and *Executor Actor Role* is overlapping, which means that some instances of *Transaction Kind* relate with only one instance of the *Actor Role* powertype.

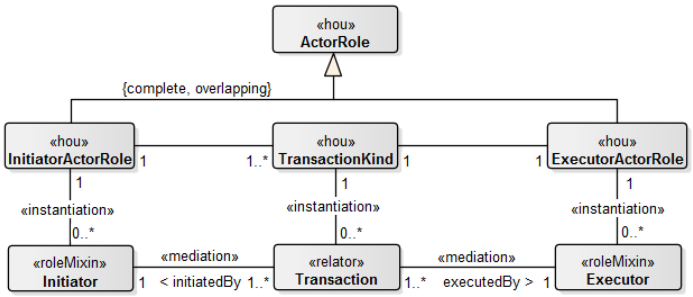


Fig. 2. Initiator and Executor Actor Roles in relation to Transaction Kind

Following DEMO, we assume that an actor playing a particular actor role in an organization commits himself to perform coordination acts and to accept social commitments of certain kinds under certain types of situations. Hereafter, social commitments resulting coordination acts are called *C-commitments* (C- for ñcoordinationò). Moreover, since social commitments and claims always appear in pairs, we refer to *C-claims* that result from coordination acts in addition to C-commitments. Note that we do not use the term ñC-factò originally proposed in DEMO as the result of coordination acts. This substitution of terms is motivated by the considerable difference in understanding of the notion of fact put forth by the UFO and the DEMO ontological commitments.

The model depicted in Fig. 3 facilitates explicit representation of C-act types, C-commitment and C-claim types which constitute a transaction. The details about relations of endurants and events can be found in [14].

Subsequent coordination acts performed by actors involved in a transaction contribute to the life of this transaction over time, i.e., to the changes it might undergo. Reified transaction types in conceptual models allow explicit representation of transaction phases and properties. A transaction phase is defined by C-commitments that constitute this transaction during a certain period of time. For instance, a transaction can be suspended until a C-commitment of a particular type is created, or it can be terminated, etc. As it is the case for all endurant types, transaction phases are represented on conceptual models as specializations of a transaction kind.

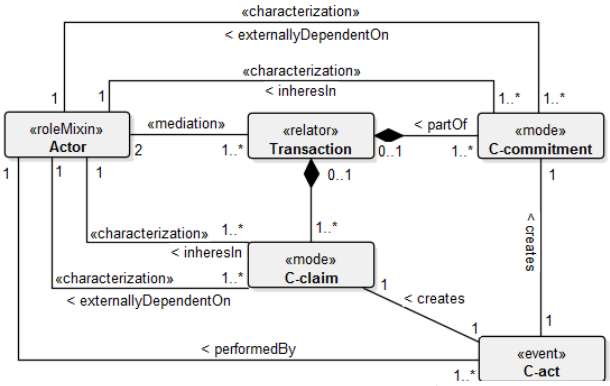


Fig. 3**.** Considering transactions as endurants composed of C-commitments and C-claims

By considering transactions as endurants, we are able to specify their qualities in addition to qualities of the participating actors. For example, a yearly membership registration (*Transaction*) of a customer (*Initiator*) in a company (*Executor*) can have a particular cost (a quality inhering in the transaction itself). Further, a modeler can characterize changes of a transaction cost over time. Another example is characterization of transactions by a status that can be, for instance, ñsuccessfulò, ñfailedò, or ñunconfirmedò.

As full-fledged endurants, transactions can play roles [15]. For instance, outsourced transactions acquire specific contingent properties being parts of organizational structures of third-party companies. Transactions initiated externally (i.e., by the initiator from the environment of the organization under consideration) can play a role of service agreements [16] provided by the organization.

4.2 Transactions as Events

According to the C9 and C10 DEMO commitments, a transaction process extends in time by accumulating its temporal parts similar to other perdurants. In [11], it is proposed to consider perdurants (there, generally termed events) as the manifestation of individual qualities and relationships. Taking into account this notion of events, a transaction process is a *Transaction Event* focusing on relationships of actors involved in a minimal business interaction (i.e., on a transaction). In this section, we elaborate on the practical relevance of having a transaction event as a modeling construct. Our arguments are based on those in [11] made in favour of reifying events as the context of relationships.

Hereafter, we consider a transaction event constituted by coordination acts of involved actors. These C-acts result in coordination commitments and claims, which, in turn, constitute the transaction in the focus of the transaction event. Taking into account that ñroles are usually understood as ways of participation[3] to an eventò [11],

[3] Following [11], we understand participation as a formal relation linking endurants and perdurants.

the C7 commitment can be reformulated as follows: *an actor is a social individual which participates a transaction event by making C-acts* (Fig. 4).

The UML composition relation (represented by an association having a black diamond in the association end connected to the class representing the whole) in Fig. 4 implies that the parts in the depicted part-whole relations are non-shareable among things of that whole class, i.e., that the maximum cardinality w.r.t. to the whole class is 1. A transaction event should be understood as an optional whole for coordination acts.

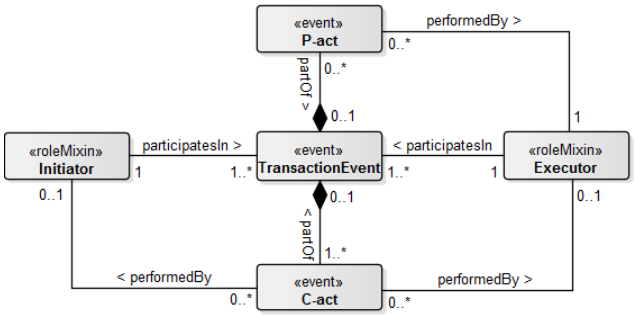


Fig. 4. Considering transactions as events composed of C- and P-acts

The execution phase of a transaction (see Sect. 2) is constituted by a production act, and it can be modeled as a proper part of a transaction event. Considering the execution phase as an event allows unambiguous relation of this phase with the completion (events) of other transactions.

A modeler may want to explicitly represent other temporal constraints concerning a transaction process. For instance, one may want to introduce a pick up event as a proper part of a delivery transaction in order to express the constraints of the transaction duration. When a transaction event evolves in time, it goes through phases composed of events. Contrary to transaction phases composed of C-commitments and C-claims (i.e., endurants), these phases are complex events. By explicitly represented events (i.e., transaction events, C-acts, and P-acts), we can represent temporal and causal relations between them [14].

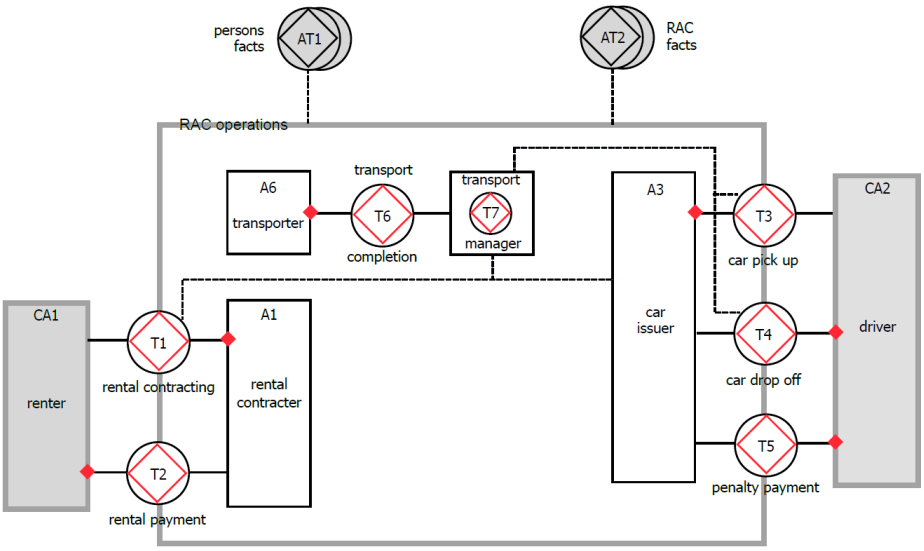
Finally, the consideration of a transaction event in a broader context of the involved objects and their qualities facilitate the specification of constraints for constituting coordination acts. For instance, a commitment for a delivery can be restricted by values of a requested drop-off location of a delivered product.

5 Applying the Transaction Pattern: a Case Study

We believe that the conceptual models and ontological distinctions proposed in previous sections facilitate understanding and communication of various organizational implementations. In [5], the authors provided a thorough analysis of organization implementation descriptions of OMGs EU-Rent case [17], using the DEMO con-

struction model of a fictitious car rental company for the representation of implementation independent organizational essence. In this section, (a part of) the analysis given in [5] is supplemented by OntoUML representations, which were obtained by specializing the modeling structures from in Sect. 4. Since we did not transform the DEMO modeling language to OntoUML, the conceptual models in this section should be considered as additions to the model depicted in Fig. 5.

The DEMO construction model (Fig. 5) representing the immutable ontological essence of a fictitious car rental company ñRent-A-Carò (of RAC for short) was borrowed from [5]. Hereafter, we elaborate on some organization implementation choices with the references to this ontological essence. A reading guide for this model can be found in [18].



Transaction kind	Product kind
T1 rental contracting	P1 Rental is contracted
T2 rental payment	P2 the rent of Rental is paid
T3 car pick up	P3 the car of Rental is picked up
T4 car drop off	P4 the car of Rental is dropped of
T5 penalty payment	P5 the penalty of Rental is paid
T6 transport completion	P6 Transport is completed
T7 transport management	P7 transport management for Day is done

Fig. 5. The Organization Construction Diagram (OCD) and the Transaction Result Table (TRT) of a car rental company (after [5])

According to the DEMO analysis, ñcar drop offò is one of the transaction kinds in RAC, of which ñcar issuerò and ñdriverò are the participating actor roles (see Fig. 5). By specializing the model depicted in Fig. 2, a modeler can explicitly express that ñcar drop offò is an instance of *Transaction Kind*, and instances of the *Car Drop Off*

type are transactions (Fig. 6); actors playing the *Car Issuer* role initiate *Car Drop Off* transactions, while actors playing the *Driver* role execute them. In Fig. 6 and other figures of this section, we highlight the elements of the models from Sect. 4 in grey.

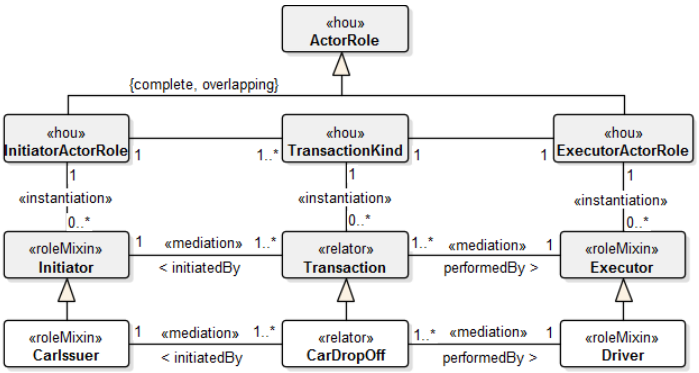


Fig. 6. *Car Drop Off* transactions as objectified social relationships

Case description in [5] provides further details about *Car Drop Off* transactions. The RAC company allows cars to be dropped-off in different locations. Students are hired to implement this service: ***ñ****For a small amount of money, a student would await the arrival of a rented car, e.g. at an airport, and drive it back to the office of RAC, after which the student would go home by public transport****ò*** [5].

The given implementation of the company was analyzed in [5] as follows. ***ñé*** *Students are authorized to accept the drop-off, so there is an assignment between employees and act types (during some time frame), and, as the student is not the requester of the drop-off, there is some form of delegation****é ò*** [5]. Some organization implementation variables were extracted from this description including V1[4] and V2:

- V1: Cross-reference which employee is allowed to perform which type of act (cross reference functionary type/act type);
- V2: Delegation of act types.

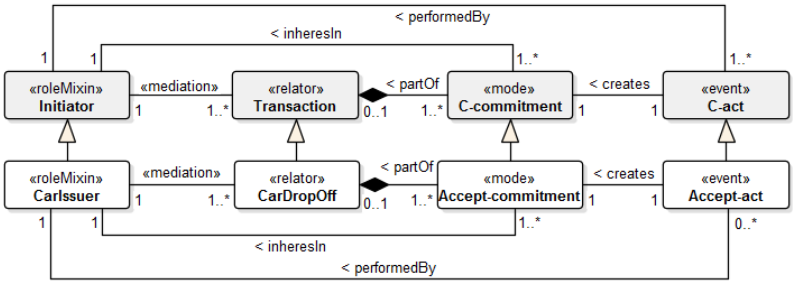


Fig. 7. *Car Issuer* participations in *Car Drop Off* transactions

[4] Here, we do not support the original numbering of implementation variables from [5].

In order to model the values of these variables actual for RAC, we explicitly represent a type of C-commitments (*Accept-commitment*), which are created when drop-offs are accepted in the scope of *Car Drop Off* transactions (Fig. 7). In this specification, the acceptance of a drop-off may not happen (in case the transaction has been failed at some moment). Although instances of *Car Issuer* are recognized as bearers of C-commitments of the *Accept-commitment* type, the execution of related C-acts can be delegated to students. As illustrated in Fig. 8, *Accept-act* C-acts are performed by instances of *Student Qua Car Issuer*, which participate in *Car Drop Off Event* transaction events.

Without further elaboration on a relation between actor roles and organizational roles (like employee, student, or director), we specify the delegation relationship between *Car Issuer* and the *Student Qua Car Issuer* organizational role, where the latter can be thought of as a specialization of the *Student* type. Contrary to *Car Issuer*, instances of the *Student Qua Car Issuer* type are identifiable persons. This fact is represented by the *role* stereotype and the kind *Person*, which is a supertype of this role. Despite participating in car drop off events (instances of the *Car Drop Off Event* type), instances of *Student Qua Car Issuer* do not bear social commitments created by accepting coordination acts.

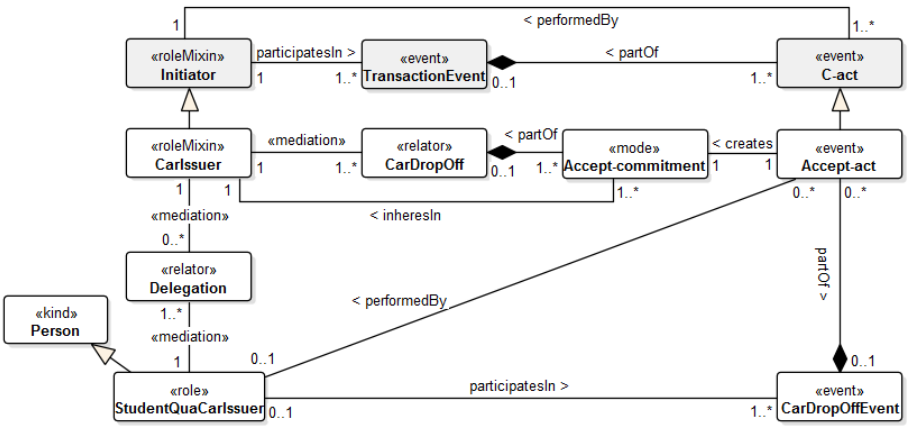


Fig. 8**.** Accepting coordination acts of *Car Drop Off* transactions delegated to students

As noted in [5], ***ñé*** *the drop-off location could be anywhere (airport departure hall 3, town center,* ***é*** *) and not necessarily a RAC office. This implies that the state and accept of the drop-off can happen at any location. For that, the locations of performing certain acts must be defined.****ò*** Another organization implementation variable was defined accordingly as follows:

Þ V3: Cross-reference which act type can be performed in which location (event location restrictions).

The behavioural aspects of the transaction pattern can be referred to for grounding the discussions on values of this implementation variable. Obtained by specializing

the model in Fig. 4, the lower half of the model in Fig. 9 expresses a constraint concerning the location where the acceptance of a dropped car (an instance of *Car Dropped Off-Accept*) can be performed by an actor playing the *Car Issuer* actor role. The location constraint cannot be applied to the *Car Drop Off* type (see Fig. 5), since a location is not directly involved in transactions [11].

The upper and the lower parts of the model in Fig. 9 together illustrate the interdependency of the location constraints of C-acts constituting transaction events of different types. Based on this model, an enterprise modeler can further specify to which extent the execution of C-acts of a particular type is restricted to the location at which the whole transaction event is triggered. This interdependency can further be specified, e.g., ñ*pick-up can only be done at branches near airports, while drop-off can be done at any branch*ò [5]. For the given implementation of RAC, the *CarPickUp-Promise* event can be constrained by *Airport Area*, which is a specialization of *Location.*

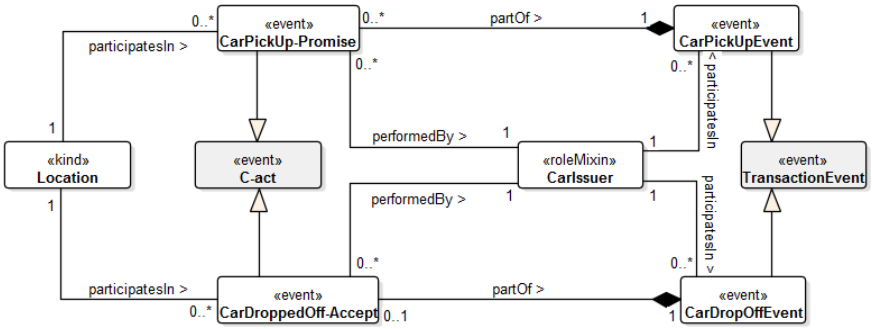


Fig. 9**.** Location constraints of C-acts

One can imagine the implementation of RAC, in which the acceptance of a car by a student playing the *Student Qua Car Issuer* actor role triggers the planning of a maintenance control required from a mechanical engineer, i.e., a *Transport Completion* transaction (see Fig. 5).

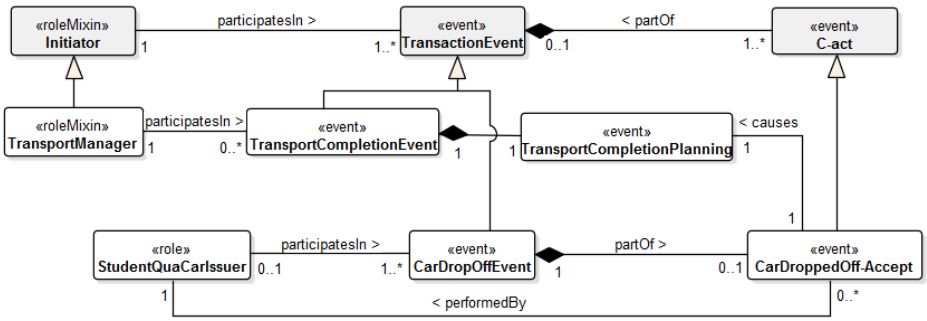


Fig. 10**.** Interrelation of *Car Drop Off* and *Transport Completion* transactions

The notion of transaction event from Sect. 4 allows referring to a transport completion transaction before a transport manager has initiated it. In its planning stage, a transport completion transaction can be represented by a transport completion transaction event (Fig. 10).

6 Final Considerations

Enterprise conceptual modeling is not an easy task. In this paper, we make an attempt to prepare foundations for possible extensions of the DEMO-based conceptual models by considering the OntoUML representation of the DEMO transaction pattern. By using the proposed modeling constructs, we represent some implementation aspects of a fictitious Rent-A-Car company and demonstrate the ability of these constructs to forth making important ontological distinctions that can be overlooked in ordinary textual descriptions of implementations. We also demonstrate the benefits of the incorporation of events into structural conceptual models.

Acknowledgments. This research is partly funded by the FEDER grant number HN0002134: CLASSE 2 (ñLes Corridors Logistiques: Application a la Vallee de la Seine et son Environnementò). This research is also partly funded by the Brazilian Research Funding Agencies CNPq (grants number 311313/2014-0 and 461777/2014-2) and FAPES (grant number 69382549).

References

1. Wand, Y., Weber, R.: Research Commentary: Information Systems and Conceptual Modeling ï A Research Agenda. Information Systems Research, Vol. 13, Issue 4, pp. 363ï 376. INFORMS, USA (2002)
2. Barjis, J.: Enterprise Modeling and Simulation Within Enterprise Engineering. Journal of Enterprise Transformation, 1:3, pp. 185-207. Taylor&Francis Online (2011)
3. Dietz, J.L.G.: Enterprise Ontology ï Theory and Methodology. Springer-Verlag, Berlin Heidelberg (2006)
4. Guizzardi, G.: Ontological foundations for structural conceptual models. Telematics Instituut Fundamental Research Series, ISSN 1388-1795, No. 015, The Netherlands (2005).
5. Opđ Land, M., Krouwel, M.: Exploring Organizational Implementation Fundamentals. In: Proper , H.A., Aveiro, D., Gaaloul, K. (eds.) Advances in Enterprise Engineering VII (EEWC 2013). LNBIP, Vol. 146, pp. 28-42. Springer, Berlin Heidelberg (2013)
6. Guizzardi, G., Wagner, G., Almeida, J.P.A., Guizzardi, R.S.S.: Towards Ontological Foundation for Conceptual Modeling: The Unified Foundational Ontology (UFO) Story. Applied Ontology, Vol. 10, Issues 3-4, pp. 259-271. IOS Press (2015)
7. Verdonck, M., Gailly, F.: Insights on the Use and Application of Ontology and Conceptual-Modeling Languages in Ontology-Driven Conceptual Modeling. In: 32th International Conference ER 2016. LNCS, Vol. 9974, 83-97. Springer International Publishing (2016)
8. Habermas, J.: The theory of communicative action. Lifeworld and system: a critique of functionalist reason. Vol. 2. (Translated by Thomas McCarthy), 3d corrected edition 1985. Suhrkamp Verlag, Frankfurt am Main (1981)

9. Dietz, J.L.G.: The PSI theory ï understanding human collaboration. Technical Report TR-FIT-15-05. Faculty of Information Technology Czech Technical University in Prague (2015) [online, capture in 2016: http://www.ciaonetwork.org/uploads/eewc2015/ee_theories/theories/]
10. Guizzardi, G., Falbo, R.A., Guizzardi, R.S.S.: Grounding software domain ontologies in the Unified Foundational Ontology (UFO): the case of the ODE software process ontology. In: XI Iberoamerican Workshop on Requirements Engineering and Software Environments, pp. 244ï 251 (2008)
11. Guarino, N., Guizzardi, G.: Relationships and Events: Towards a General Theory of Reification and Truthmaking. In: Adorni G., Cagnoni, S., Gori, M., Maratea, M. (eds.) 15th International Conference of the Italian Association for Artificial Intelligence (AI*IA 2016). LNAI, Vol. 10037, pp. 237-249. Springer International Publishing (2016)
12. Carvalho, V.A., Almeida, J.P.A., Guizzardi, G.: Using a Well-Founded Multi-Level Theory to Support the Analysis and Representation of the Powertype Pattern in Conceptual Modeling. In: Nurcan S., Soffer P., Bajec M., Eder J. (eds.) Advanced Information Systems Engineering (CAiSE 2016). LNCS, Vol. 9694, pp. 309-324. Springer, Cham (2016)
13. Falbo, R.A., Ruy, F.B., Guizzardi, G., Barcellos, M.P., and Almeida, J.P.A.: Towards an Enterprise Ontology Pattern Language. In: 29th Annual ACM Symposium on Applied Computing, pp. 323ï 330. ACM (2014)
14. Guizzardi, G., Wagner, G., De Almeida Falbo, R., Guizzardi, R.S.S., Almeida, J.P.A.: Towards Ontological Foundations for the Conceptual Modeling of Events. In: 32th International Conference ER 2013. LNCS, Vol. 8217, 327-341. Springer Berlin Heidelberg (2013)
15. Guarino, N., Guizzardi, G.: ñWe Need to Discuss the Relationshipò: Revisiting Relationships as Modeling Constructs. In: 27th International Conference, CAiSE 2015. LNCS, Vol. 9097, pp. 279-294. Springer International Publishing (2015)
16. Nardi, J.C., De Almeida Falbo, R., Almeida, J.P.A., Guizzardi, G., Ferreira Pires, L., Van Sinderen, M.J., Guarino, N., Fonseca, C.M.: A Commitment-based Reference Ontology for Services. Information Systems, Vol. 54, pp. 263-288. Elsevier Ltd. (2015)
17. Object Management Group: Business Motivation Model (BMM) Specification, V1.1. OMG Available Specification OMG Document Number: formal/2010-05-01 (May 2010) http://www.omg.org/spec/BMM/1.1/PDF/
18. Op 't Land, M., Dietz, J.L.G.: Benefits of Enterprise Ontology in Governing Complex Enterprise Transformations. In: Albani, A., Aveiro, D., Barjis, J. (eds.) Advances in Enterprise Engineering VI (EEWC 2012). LNBIP, Vol. 110, pp. 77-92. Springer, Heidelberg (2012)