Proceedings of the Federated Conference on Computer Science and Information Systems pp. 1153–1158
DOI: 10.15439/2016F145
ACSIS, Vol. 8. ISSN 2300-5963

Towards Paired Transactions Modeling

Frantisek Hunka
University of Ostrava, Faculty of Science
Dvorakova 7, 701 03 Ostrava
Czech Republic
Email: frantisek.hunka@osu.cz

Jiri Matula
University of Ostrava, Faculty of Science
Dvorakova 7, 701 03 Ostrava
Czech Republic
Email: jiri.matula@osu.cz

Abstract—**Paired transactions or paired transfers have their origin in accountancy systems. The Resource-Event-Agent (REA) ontology uses paired transactions as a basic building block for business process modeling. A business process (REA model) is composed of two sets of paired transactions. REA itself originates from accountancy systems and has gradually developed into a full-fledged information system framework. REA model used to be depicted by ER diagrams and later by UML class diagrams. However, both these diagrams were not designed to capture conceptual models which are more precise, comprehensible for domain experts and easily to modify. ORM (Object Role Modeling) represents approach to conceptual modeling which fulfils above mentioned requirements. The main goal of the paper is to derive and describe the ORM model of paired transactions which corresponds to REA exchange model and assess this approach.**

I. Introduction

The REA ontology can be classified as a domain specific ontology that is focused on value modeling of business processes. The three core REA concepts are *resource*, *event* and *agent* from which the name of the modeling approach was derived. The aim of the REA modeling approach is to record any changes in property rights to resources, register resource usage, resource consumption or resource production, see [1,4,7].

Resource entities can be exchanged for other resource entities in REA exchange processes and can be converted to other resources in REA conversion processes as well. A REA application keeps track of which resources were exchanged for which ones or which resources were converted for which others.

The REA model records information based on the coherence between data of one or more business events. The REA process is defined by related REA events and has at least two composite economic events: a *decrement event* that outflows, consumes or uses the outgoing resource(s) and an *increment event* that inflows or produces the incoming resource(s). The REA process is called the REA model and represents the notion of a business process.

The main benefit of the REA modeling approach is the possibility of keeping track of primary and raw data about economic resources, by [5]. All accounting artifacts such as debit, credit, journals, ledgers, receivables and account balances are derived from the data describing exchange and conversion REA processes [4,8]. For example, the data describing the sale event is used in the warehouse management, payroll, distribution, finance and other application areas, without transformations or adjustments.

The quality of a database application depends crucially on its design, see [13]. To ensure correctness, clarity, adaptability and productivity, information systems should be specified at the conceptual level first, using concepts and the language that both designers and customers can easily understand [6]. Object-Role-Modeling (ORM) is a fact oriented approach for modeling information at the conceptual level. A fact is a particular arrangement of one or more objects. Depending on the number of objects that are involved in a fact, we speak about unary, binary, ternary, etc., facts. An example of unary fact is that *Vendor is a Person*. Another example of binary fact is that a *Customer receives a Pizza*. Unlike traditional approaches, ORM make no use of attributes as a base constructs, instead expressing all facts types as relationships [6]. This attributes free-approach leads to greater semantic stability in conceptual models and enables ORM fact structure to be directly verbalized and populated using natural language sentences.

The ORM method provides a more precise way to capture and validate data concepts and business rules with domain experts. ORM diagrams simply capture the world in terms of objects (entities or values) that play roles (parts in relationships) which forms a fact. ER notation as well as UML notation allows relationships to be modeled as attributes. ORM models the world in terms of objects and roles, and hence has only one data structure – the relationship type. As a consequence, ORM diagrams take up more room than corresponding UML or ER diagrams. The aim of the paper is to find out a "semantic" connection between REA business process model and fact-based model utilizing ORM approach.

The structure of the paper is as follows. Section Two describes REA modeling approach. Concise possibilities of the ORM modeling method are mentioned in Section Three.

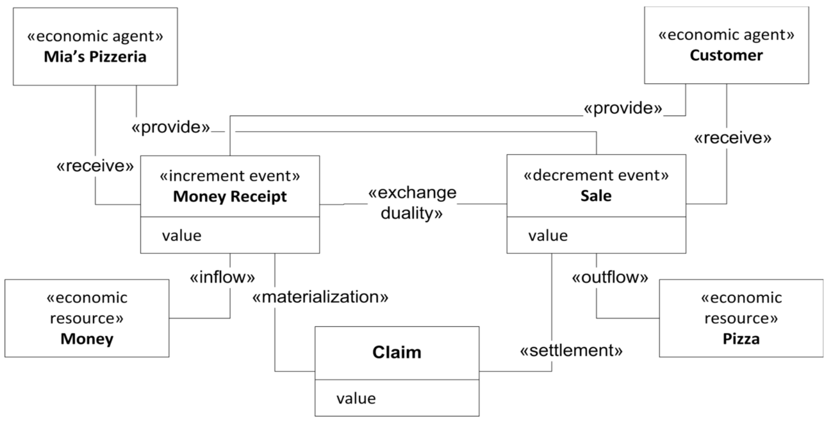

Fig. 1 REA core pattern example

ORM model of paired transactions is described and illustrated in Section Four. Discussion of the results is mentioned in Section Five and conclusion is contained in Section Six.

II. REA Modeling Approach

The REA ontology is based on the REA core pattern that expresses the basic principle [2,4]. The fundamental entities of the pattern are different events that are involved in various transactions which have something in common; there has always been a *decrement economic event* (one in which something is provided) and an *increment economic event* (one in which something is received). Apart from economic events, the economic agents that represent human beings partake in the exchange process. Resources are entities which are kept track because their property rights can be exchanged or they can be converted to create new resources. As the mutually bound events cannot happen at the same time, the claim entity is utilized for deferred revenue, prepaid expenses, accounts payable and so on. The REA core pattern is illustrated as a Pizzeria shop in Fig. 1. The economic events in REA models usually encapsulate properties for *date, time* and *location* in space.

The REA model is an extension of the REA core pattern. The principal feature of the REA modeling approach is that it explicitly distinguishes between past and current events and events performed in the future for which it introduces the commitment entity. The relationships of *committed provide* and *committed receive* mean that some agreement about the future exchange has to be achieved between economic agents. The commitment entity addresses the issue of modeling promises of future economic events and the issue of reservation of resources. Commitment entities and their relationships with other entities are shown in Fig. 2. To a considerable extent, the commitment entity copies the structure of the event entity, by which we mean the existence of an increment and decrement commitment and the exchange reciprocity relationship. The exchange reciprocity relationship between the increment and decrement commitments identifies which resources are promised to be exchanged for which other resources.

Each commitment is related to an economic resource by a reservation relationship which specifies which resources will be needed or expected by future economic events. The reservation relationship between the resource and commitment represents obligation of economic agents to provide or receive rights to economic resources in exchange processes and represents scheduled usage, consumption or production of economic resources in conversion processes.

The most important relationships of the REA model are the *exchange reciprocity* and *exchange duality* relationships, by [5,9,10,11]. The exchange reciprocity relates a pair of an increment and decrement commitment entities. The exchange reciprocity relationship identifies which resources are promised to be exchanged for which others.

The exchange duality relationship which relates corresponding increment and decrement economic events keeps track of which resources were exchanged for which ones.

III. ORM Conceptual Modeling Method

Object-Role Modeling is a conceptual modeling method that views the world as a set of objects that play roles (parts in relationships) according to [6]. For example, you may play a role of walking in the country (a unary relationship involving just you) or you may play a role reading this paper (a binary relationship between you and the paper). Thus a role in ORM corresponds to an association-end in UML, except that ORM also allows unary relationships. Object-Role Modeling is a conceptual modeling method that views the world as a set of objects that play roles (parts in relationships) according to [6].

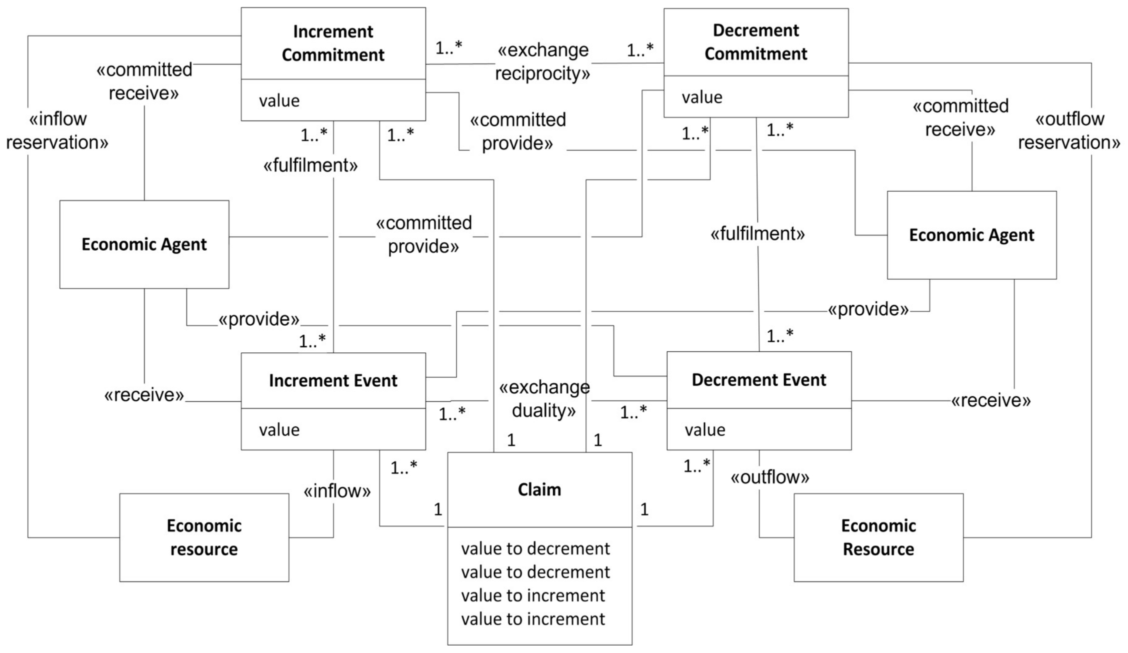

Fig. 2 REA model. Adapted from [1]

The main structural difference between ORM and UML is that ORM excludes attributes as a base construct and treats them instead as a derived concept. The conceptual schema using ORM specifies the information structure of the application in the forms of: *fact types* that are of interest; *constraints* on these; and *derivation rules* for deriving some other facts.

A fact is a proposition that is taken to be true by the relevant business community. A fact type is a kind of fact that may be represented in the database [3]. The constraints represent constraints or restrictions on populations of the fact types. The derivation rules include rules that may be used to derive new facts from other facts, see [6,12].

The ORM model (left part of Fig. 3) indicates that employees are identified by their employee numbers. The top three roles (*EmpName*, *Title* and *Sex*) are mandatory roles. This is indicated by the black dots at the *Employee* box. The other black dot where two roles are connected (at the bottom of *Employee*) is a disjunctive mandatory role constraint indicates that an employee must have a social security number or a passport number or both. The uniqueness of constraints (cardinalities in UML) indicates vertical lines over roles. In Fig. 3 it means that *empNr*, *EmpName*, *Sex*,

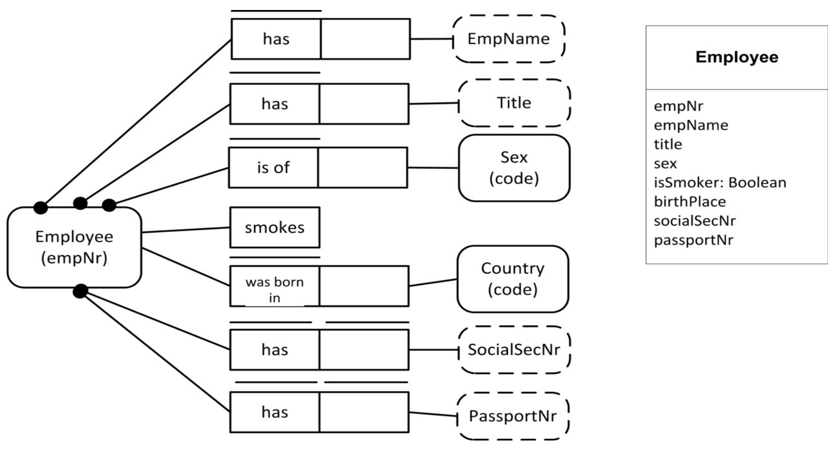

Fig 3. ORM and UML models of Employee

and *Country* is unique for each employee. Two vertical lines over each roles (*SocialSecNr*, *PassportNr*) indicating that each employee number, social security number and passport number refers to the one employee at most. The dashed line over e.g. *PassportNr* indicates that this is a value not an object.

Graphically, object types are depicted as named boxes (solid for entity types, and dotted for value types). As in logic, a predicate is a proposition with object-holes in it. In ORM, a predicate is treated as an ordered set of one or more roles, each of which is depicted as a box, which may optionally be named. A fact type is formed by applying a predicate to the object types that play its roles.

IV. ORM Model of Paired Transactions

The ORM model of paired transactions is composed of two kinds of transactions (left hand side, right hand side). These paired transactions actually represent result of an exchange process in which some resources were exchanged for other resources. The cardinality between exchanged resources is in general many-to-many as was stated in the REA model, see Fig. 2. Despite the REA model, the ORM model of paired transactions contains only one relationship that joins both kinds of transactions which is called the duality relationship. The name *duality* expresses the final state of an exchange process. The duality has mandatory relationships to both kinds of transactions. The left-hand side transactions represent a transfer of goods and the right-hand transactions stand for a transfer of money. We consider the common case of transfer in which resources or services are exchanged for money. These two transfers are depicted by object classes with the same name. The object class *Goods Transfer* is related to two kinds of actor roles the *vendor* and the *customer* who both belong to the object class *Person*. There is a relationship *exclusive or* between two actor's roles which means that these actor roles have to be different persons. The *customer* is an actor role who receives goods in *Goods Transfer*. The *vendor* is an actor role who provides the goods for the transfer. The corresponding object class *Money Transfer* is related to the *payer* and the *cashier* actor's roles. Between both actor roles there is a relationship *exclusive or* with the same meaning as in the previous case. Both *payer* and *cashier* belong to the object class *Person*. It is important to use a proper actor role. The *customer* can be e.g. wife and the *payer* can be her husband.

The object class *Goods Transfer Contracted* is a subclass of the object class *Goods Transfer*. This relationship between these object classes means that there must exists at first *Goods Transfer* object class and subsequently it may happen that the object class *Goods Transfer Contracted* becomes existent. Conversely, *Goods Transfer Contracted* cannot exist without the object class *Good Transfer*. This point is very important and differs from the REA model. The object class *Goods Transfer Contracted* means that the *customer* promised that he would receive the goods and the *vendor* promised that he would deliver the goods. But as we are talking about paired transactions there must be the other transaction(s) because paired transactions mean that some resources are transferred in consideration of the other resource transfers.

The other transaction(s) of the paired transactions is represented by the object class *Money Transfer Contracted*. In this case, the *payer* promised that he would pay for the goods and the *cashier* promised that he would accept the amount of money (resource). At this point it is essential that the state *promise* is reached on both object classes *Goods Transfer Contracted* and *Money Transfer Contracted*.

Each object class (*Goods Transfer Contracted*, *Money Transfer Contracted*) contains property types of contracted goods kind and contracted location. The object class *Money Transfer Contracted* has the same kinds of property attributes and scale attributes. These property attributes and scale attributes represent facts that were contracted. The construction of this model enables that the resource kind can occur more times. For instance a *customer* would like to buy the given number of a specific pizza types, a certain number of cola kinds and a certain number of chocolate kinds. In general, *Money Transfer Contracted* may represent some kind of payment before the purchase, payment after delivery and possibly payment in installments.

Goods Transfer Completed is the next object class that is a subclass of *Goods Transfer Contracted* object class. The meaning of the subclass relationship is as follows: a transfer have to be contracted at first and then it can be completed. The property attributes and the scale attributes are the same as in *Goods Transfer Contracted* but in this case the property attributes express the real values. In the detail insight, the proposed solution enables differences between the individual number of *Goods Transfer Completed* and the number of *Goods Transfer Contracted*.

Goods Transfer Contracted represents planned transactions. The delivery, which is performed in *Goods Transfer Completed* is usually performed in several shipments. The corresponding object class to *Goods Transfer Completed* is the object class *Money Transfer Completed*. This object class property attributes deal with real payment transactions which means that they deal with the real installments. The business rules are stated in the contract which comes into existence when the transactions are contracted. The structure of the property attributes is the same as the structure of *Money Transfer Contracted* but the real values may be different. All this recorded information is required in database solutions.

V. Discussion

Designing an information system involves building a formal model of the application domain which requires a good understanding of the application domain and utiliziation of the proper tools for modeling specifications in a clear and unambiguous way. ORM simplifies the design

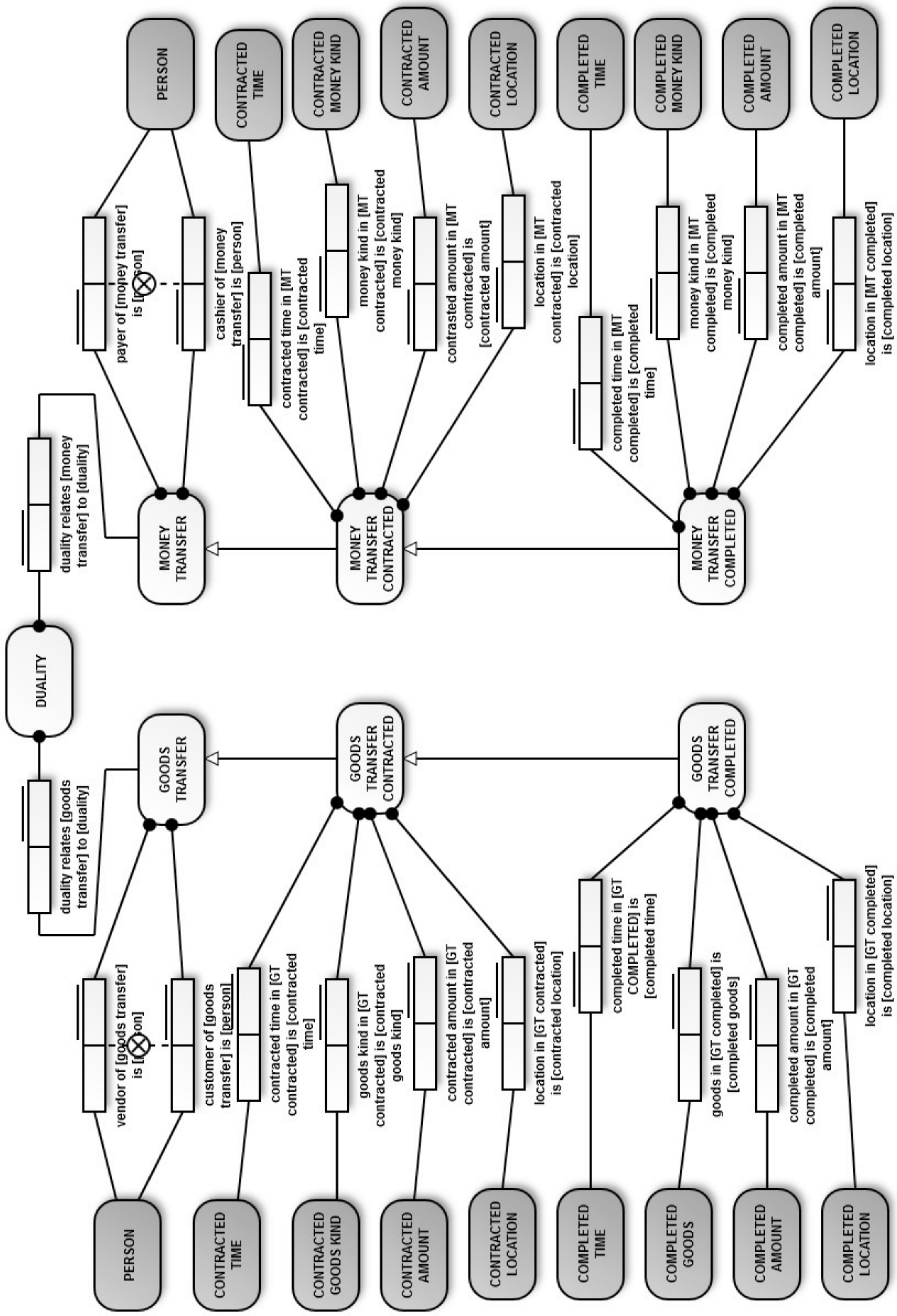
PERSON
CONTRACTED TIME
CONTRACTED MONEY KIND
CONTRACTED AMOUNT
CONTRACTED LOCATION
COMPLETED TIME
COMPLETED MONEY KIND
COMPLETED AMOUNT
COMPLETED LOCATION
payer of [money transfer] is [person]
cashier of [money transfer] is [person]
contracted time in [MT contracted] is [contracted time]
money kind in [MT contracted] is [contracted money kind]
contrasted amount in [MT comtracted] is [contracted amount]
location in [MT contracted] is [contracted location]
completed time in [MT completed] is [completed time]
money kind in [MT completed] is [completed money kind]
completed amount in [MT completed] is [completed amount]
location in [MT completed] is [completed location]
duality relates [money transfer] to [duality]
MONEY TRANSFER
MONEY TRANSFER CONTRACTED
MONEY TRANSFER COMPLETED
DUALITY
GOODS TRANSFER
GOODS TRANSFER CONTRACTED
GOODS TRANSFER COMPLETED
duality relates [goods transfer] to [duality]
vendor of [goods transfer] is [person]
customer of [goods transfer] is [person]
contracted time in [GT contracted] is [contracted time]
goods kind in [GT contracted] is [contracted goods kind]
contracted amount in [GT contracted] is [contracted amount]
location in [GT contracted] is [contracted location]
completed time in [GT COMPLETED] is [completed time]
goods in [GT completed] is [completed goods]
completed amount in [GT completed] is [completed amount]
location in [GT completed] is [completed location]
PERSON
CONTRACTED TIME
CONTRACTED GOODS KIND
CONTRACTED AMOUNT
CONTRACTED LOCATION
COMPLETED TIME
COMPLETED GOODS
COMPLETED AMOUNT
COMPLETED LOCATION

Fig. 4 ORM model of paired transactions

process by using natural language, and by examining the information in terms of simple and elementary facts. By expressing the model in terms of natural concepts, like objects and roles, it provides a conceptual approach to modeling.

The REA modeling approach addresses the paired transactions model in the REA core pattern and in the REA model. The REA core pattern corresponds to accounting model and captures events which were completed. The REA model is more general than the REA core pattern and provides possibilities to address future events. However, connection between the commitment entity and the event entity in the REA model is insufficiently consistent. This is done by the fact that both commitment and event entities represent production and that the REA model doesn't have a proper state machine, see [9]. For these reasons it is difficult to distinguish the phases in which the paired transactions were *contracted* (promised) and *completed*. When comparing the REA core pattern with the REA model it is evident that the REA model covers all operations of the REA core pattern despite the fact that the commitment actions are only formal.

The ORM model of the paired transactions enables clear distinguishing between the *contracted phase* and the *completed phase*. This is done by utilization a sub-classing mechanism of the ORM modeling approach. The corresponding object classes *Goods Transfer Contracted* and *Money Transfer Contracted* express the *contracted* (planned) property types and attributes types which are essential to be stored in the database solution. In the same way, *Goods Transfer Completed* and *Money Transfer Completed* capture the real value of property types and attribute types of finished paired transactions.

The object classes *Goods Transfer* and *Money Transfer* form the beginning of the paired transactions process which means that they identify partaking actor roles which will be involved in the process. They have to be created first. After that, the object classes *Good Transfer Contracted* and *Money Transfer Contracted* can be created. Similarly, existence of *Goods Transfer Completed* and *Money Transfer Completed* is dependent on the existence of contracted transfers.

VI. Conclusion

The paper deals with paired transactions modeling utilizing the ORM conceptual modeling method. The benefits of this mathod can be sumarized as follows. This method and modeling approach enables to explicitly distinguish the contracted phase from the completed phase of the paired transactions model. It also ensures unified transaction processing which means utilizing only contracted and completed phases. The proposed solution also eliminates usage of the claim temporal entity. Future research will cover implementation, verification and validation of the proposed ORM model.

Acknowledgement

The paper was supported by the grant provided by Ministry of Education, Youth and Sport Czech Republic, reference no. SGS15/PRF2016.

References

[1] G. L. Geerts, and W. E. McCarthy, “The Ontological Foundation of REA Enterprise Information Systems”. Paper presented at the Annual Meeting of the American Accounting Association, Philadelphia, PA., 2000.

[2] McCarthy, W.E. “The REA Accounting Model: A Generalized Framework for Accounting Systems in A Shared Data Environment.” The Accounting Review (July) 1982, pp. 554-578.

[3] J. L. G. Dietz, “The Essence of Organization. An Introduction to Enterprise Engineering”. Sapio bv, 2012.

[4] Ch. L. Dunn, O. J. Cherrington, and A. S. Hollander, Enterprise Information Systems: A Pattern Based Approach. New York: McGraw-Hill/Irwin, 2004.

[5] Hruby P., Model-Driven Design Using Business Patterns. Springer-Verlag Berlin Heidelberg, 2006.

[6] Halpin, T. A., Morgan, T. Information Modeling and Relational Databases. San Francisco: MorganKaufmann, 2nd edt 2008.

[7] Dudycz, H., Korczak, J. “Conceptual design of financial ontology”, Proceedings of the 2015 Federated Conference on Computer Science and Information Systems. pp. 1505-1511. DOI: 10.15439/2015F162

[8] F. Hunka, and J. Zacek, “Detailed Analysis of REA Ontology”, Lecture Notes in Busines Information Processing, Vol. 174, 2014, pp. 61-75. DOI: 10.1007/978-3-319-06505-2

[9] Hunka, F., Zacek, J. A new view of REA state machine (2015) in Applied Ontology, 2015, vol. 10 (1), pp. 25-39.

[10] R. Klimek and P. Szwed, “Verification of ArchiMate Process Specification Based on Deductive Temporal Reasoning”. Proceedings of the 2013 Federated Conference an Computer Science and Information Systems. pp. 1103-1110.

[11] Korczak, J., Dudycz, H., Dyczkowski, M. “Design of Financial Knowledge in Dashboard for SME Managers”. Proceedings of the 2013 Federated Conference an Computer Science and Information Systems. pp. 1111-1118.

[12] Kersten, G., Wachowicz, T. “On Winners and Losers in Procurement Auctions”. Proceedings of the 2014 Federated Conference an Computer Science and Information Systems. pp. 1163-1170. DOI: 10.15439/2014F271.

[13] Paweloszek, I. “Approach to Analysis and Assessment of ERP System. A Software Vendor's Perspective”. Proceedings of the 2015 Federated Conference an Computer Science and Information Systems. pp. 1415-1426. DOI: 10.15439/2015F251.