

# Modeling Issues in REA

Anders Hessellund

Software Development Group  
IT University of Copenhagen, Denmark  
hessellund@itu.dk  
<http://www.itu.dk/people/hessellund/>

**Abstract.** Enterprise information systems pose a serious challenge to our traditional modeling schemes. Domain experts and software developers must communicate and collaborate successfully in order to implement these systems. This process requires models that can be understood by both groups. In this paper, we examine the Resource Event Agent (REA) metamodel with respect to this challenge. Our claim is that REA can become an *ubiquitous language* which is shared by both groups if the model is specified in greater detail. We present a set of concrete modeling issues that must be resolved in order to realize this vision.

## 1 Introduction

The design and implementation of enterprise information systems requires powerful modeling techniques. Systems in this area must encapsulate sophisticated domain knowledge, support rapidly changing business requirements and provide a consistent set of views that facilitates advanced reporting. In order to manage this inherent complexity, domain experts and software developers must be able to successfully communicate about shared goals. Unfortunately, domain experts and software developers belong to different communities with their own idiosyncratic concepts, models and methods. A common medium which can be understood by both communities and therefore facilitate communication is required, if their efforts are to converge in a constructive manner.

The Resource-Event-Agent (REA) metamodel originated in the accounting domain but has later evolved into a full enterprise ontology [1–4]. REA offers domain experts with a set of atomic modeling entities which can be composed to form intricate business models. The REA entities can at the same time be understood by software developers, and REA business models can therefore be operationalized in actual software systems. These characteristics makes REA a possible medium for the communication between two very diverse communities. It is our claim that the REA ontology can serve as an *ubiquitous language* [5] for these groups, i.e., a language that both parties understand and share. The vocabulary of an ubiquitous language can be considered a *boundary object* [6]. The semantics of this boundary object can be shared and negotiated between the two groups and hence support collaborative efforts.

There are, however, still unresolved issues that need to be settled before the REA ontology can realize this vision. The semantics of the basic REA entities must be specified such that ambiguities can be eliminated. The limitations of REA implementation technologies must be explored such that REA models can be adapted to a given technological platform. The transition from design to implementation should only require a minimum of interpretation for the software developer. In short, if the REA ontology is to serve as an *ubiquitous language* then this language must be defined and documented.

In this paper, we will first describe our interpretation of REA. This interpretation relies on existing literature on REA and patterns in general. Using this interpretation as a foundation, we will sketch a set of unresolved modeling issues in the REA ontology and attempt to provide some preliminary answers and ideas. It should be emphasized that this paper presents a work in progress rather than a coherent REA modeling scheme. The paper is split in two main sections. Section 2 describes our interpretation of the REA ontology, and section 3 discuss a set of modeling issues that must be resolved. Finally, we will summarize our ideas in the conclusion in section 4.

## 2 The Resource-Event-Agent Ontology

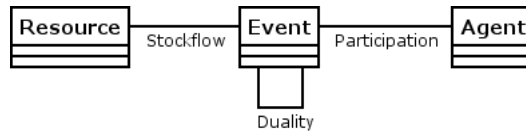
In this section, we will present the REA ontology that was introduced by William E. McCarthy in 1982 [1] and later extended in various ways, e.g., [2, 3, 7–10]. We will first explain the basic metamodel, then the suggested extensions and finally the full ontology. The purpose of describing the REA ontology is to establish our interpretation of REA as a common frame of reference with the reader.

REA has primarily been used to model accounting phenomena in information systems. It has later been discovered to be well-suited for Enterprise Resource Planning (ERP) systems where it provides a simple but generic organizing principle for the operational data of an enterprise. Traditional modeling schemes, such as those found in double-entry systems, are often accounting-specific and are therefore of little use to non-accountants. The generic and fine-grained nature of REA allows both accountants and people from other domains to share operational data and create useful reports [1].

### 2.1 The Basic REA Metamodel

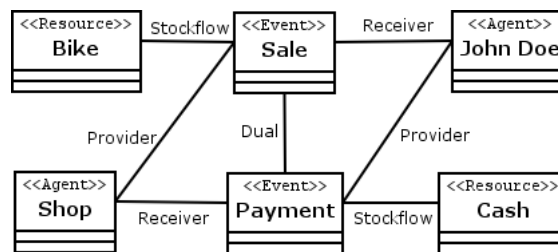
The original and basic REA metamodel consist of three basic entities and a set of relations between these entities. *Resources* are goods, services and other items that can be bought or sold. *Agents* are individuals, departments or companies that can act as sellers or buyers of resources. *Events* are the concrete acts of selling, buying or in other ways exchanging resources. These entities are connected by *stockflow*, *participation* and *duality* relations as figure 1 shows.

The common rationale in any economic transaction is that two agents agree to give each other a resource in exchange for another. An exchange is basically a pair of dual events. This economic rationale is expressed in the duality relation



**Fig. 1.** The Resource-Event-Agent metamodel

that connects the act of giving, *decrement event*, with the act of taking, *increment event*. If there is no duality then the transaction is pointless. When each part of the exchange occurs, a stockflow relation is established representing the flow of goods. Finally the participation relation between an agent and an event signifies the legal involvement of an agent in a certain transaction.



**Fig. 2.** The *transfer exchange*

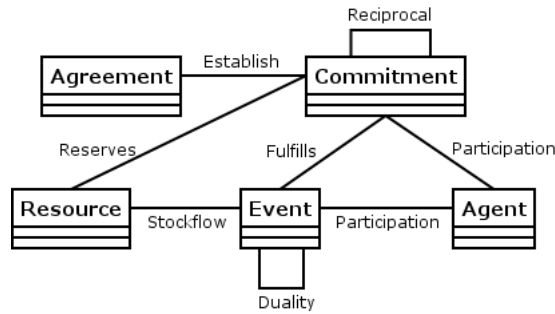
Transactions are generally captured in the *REA template* which constrains how the basic entities are connected. A basic example of the template can be seen in figure 2. This instance is called the *transfer exchange*. The *provider* and *receiver* roles of the participation will be explained in section 2.2. Other possible exchanges are possible when the entities are typed as described in section 2.3.

## 2.2 The Extended REA Metamodel

The REA metamodel has later been extended in order to encompass a greater set of business concepts. The *commitment* and *agreement* entities have been introduced by Geerts and McCarthy [2] in order to conceptualize contractual relations between business partners. These new entities are shown in figure 3.

The commitment entity has structural similarities with events. A commitment is a reified obligation to execute a certain event at a later stage. Incrementing and decrementing commitments can be *reciprocal* just like events can be dual. The relation between a commitment and an event is called the *fulfillment* relation. A commitment is connected with resources by a *reserves* relation which signifies that for instance a set of goods are to be sold in the near future. Reservations follow the basic pattern which has been proposed by Arlow

and Neustadt [11]. Agents are connected to commitments with the *participation* relation similarly to events.



**Fig. 3.** Commitment and agreement

The agreement entity organizes pairs of commitments such that complex business transactions can be ordered in a hierarchical manner. Geerts and McCarthy [2] define agreement as the reification of the reciprocal relation. However, as a business contract often contains several different transactions, it is probably more useful to define agreement simply as a set of commitments. In this manner, commitments that are not reciprocal can be combined in a single agreement. An agreement is then connected with a set of commitments with the *establish* relation.

The *location* entity was suggested by Denna et al. [3]. This entity has been described in greater detail in the UMM User Guide [7] where it used to describe where events take place. Locations are related to events by the *site* relation [7] and to commitments by the *target* relation [8] as shown in figure 4. In our previous work [8, 10], we have elaborated this concept further. The location entity is necessary in order to model business concepts from the supply chain field. An example of such a business concept is Vendor Managed Inventory (VMI) [12] which we have provided a REA model for using the location entity [8, 10]. In order to model inventories, we introduced the *capacity* relation between a location and a resource type. Capacity determines the ability of a concrete location to store instances of a certain type of resources. We will explain the idea of types in section 2.3.

Finally, the REA metamodel can also be extended in a cognitive manner as proposed by Jaquet [13]. The idea is that a concrete REA model should not necessarily be seen in the perspective of a given company. It often makes more sense to evaluate a REA model without being tied to a particular perspective. Hruby et al. [14] introduces the distinction between the *trading partner view* and the *independent view* which denotes models with or without perspectives respectively. In inter-company settings such as supply chain modeling, the

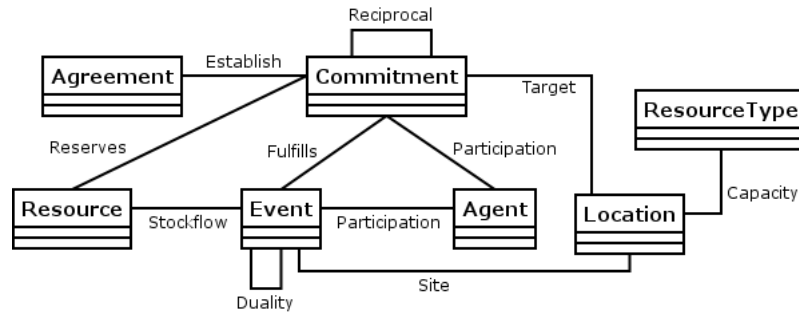


Fig. 4. Location

independent view is preferable as the models have the same meaning regardless of perspective. These models can then be used across company boundaries.

A consequence of the use of independent models is that events and commitments can no longer be characterized as decrementing or incrementing. A payment event can by its very nature be considered decrementing or incrementing depending on the perspective. The solution to this ambiguity is to distinguish between a *provider* and a *receiver* role on the participation relation [13]. The agent that provides will decrement his resources while the agent that receives will increment his resources. This set of roles enables us to view the transaction independent of perspective. The provider/receiver distinction replaces the original ambiguous distinction between *inside* and *outside* agents.

### 2.3 The Full REA Ontology

The main part of the REA metamodel's transformation into a full ontology is the introduction of typification [2, 15, 16] and a conceptual analysis based on the work of Sowa [17]. A REA ontology could potentially deliver the interoperability mechanism that Sowa asks for:

*Recent emphasis on enterprise integration ... requires shared ontologies that can support applications across all areas of business, including engineering, manufacturing, accounting, and sales.*

Sowa [17, p. 53]

Typification of the previously described REA metamodel is done by reifying the types of REA entities such that each entity has an associated type object. An event can for instance be associated with an *event type* as shown in table 1. This extends our means of expression as individual instances of these types can have varying attributes while still conforming to the general types. The typification idea was originally introduced by Fowler [18] in his distinction between *operational infrastructure* and *knowledge infrastructure*. REA entities are part of the operational infrastructure whereas types are part of the knowledge

infrastructure. The advantages of this idea are explained further by Evans [5]. A concrete example can be found in the implementation of the *adaptive object model* [19,20]. It should be noted that the relation between a type and its instances correspond to the *item-descriptor pattern* [21] rather than to *inheritance* in the object-oriented sense.

REA entity	Associated type	Examples
Resource	Resource Type	Bicycle, Cash etc.
Event	Event Type	Sale, Rental etc.
Agent	Agent Type	Company, Person etc.
Location	Location Type	Warehouse, Shop etc.
Commitment	Commitment Type	Orderline, Paymentline etc.
Agreement	Agreement Type	Contract, Schedule etc.

**Table 1.** The typification of REA

These types can be classified in further detail as shown in our previous work [8, 10] where we introduce a taxonomy for REA events and commitments. Since events and commitments have structural similarities, they can be classified according to the same scheme. A sales event and a sales commitment are for instance characterized by changing the right of ownership to a given set of resources. Similarly a produce event and a produce commitment changes the existence of a resource such that a resource appears *ex nihilo* in a production. Other events and commitments change different properties as shown in table 2.

Type	Changes ownership	Changes existence	Changes location
<b>Sale</b>	yes	no	no
<b>Payment</b>	yes	no	no
<b>Produce</b>	yes	yes	yes
<b>Use</b>	no	no	no
<b>Consume</b>	yes	yes	yes
<b>Load</b>	no	no	yes
<b>Unload</b>	no	no	yes

**Table 2.** Taxonomy for events and commitments

Table 2 is by no means exhaustive. It is possible to extend this taxonomy on both axis. We could for instance extend it on the horizontal axis by introducing the property of *usage rights*. Such a property would be useful when describing *rentals*. The taxonomy can also be extended on the vertical axis as *load* and *unload* demonstrates. These types are not present in the traditional REA literature but were introduced in our previous work [8,10] to represent the movement of goods. The taxonomy imposes a requirement on extensions to the

ontology because extensions must be fitted into this structural scheme. If this requirement is fulfilled then future extensions will be easy to compare.

We believe that it should be possible to create similar taxonomies for other REA entities. It might also be interesting to provide a description of all relations, especially what attributes these relations have. It would be a worthwhile effort since a detailed characterization of every REA entity and relation is required in order to provide rich semantics for the entire ontology. A description of the basic parts of REA will also facilitate a more precise description of different types of exchanges such as *transformation* [2] and *transport* [8, 10].

### 3 Unresolved Modeling Issues

In this section, we will discuss a set of unresolved modeling issues in our interpretation of REA. The reader should note that we consider our interpretation of REA to be sufficiently mainstream to make these issues relevant for the entire REA community. In section 3.1, we will describe the problem of balancing dual events. In section 3.2, we will discuss the issue of temporal properties of events. In section 3.3, we will sketch an idea for a possible validation concept for REA models. In section 3.4, we will bring attention to the importance of modeling compromises. Finally in section 3.5, we will criticise the lack of a role concept in REA.

#### 3.1 Balanced Duality

The duality relation has been present in the REA metamodel since the very beginning. It represents a central constraint on economic transactions, viz., every transaction must have a rationale. The presence of duality among our economic events ensures that events happen for a reason. When we deliver goods to our customers then we expect to be paid in return for the delivery. Duality pairs our decrements with increments.

Even though duality is a focal concept in REA, a central thing seems to be missing. It is not stated in the ontology, how dualities are balanced. How many increments do we need in order to match a set of decrements? If a shop delivers a bike to John Doe then how can we determine when he has participated in the right number of payment events? It does not make sense to compare the values of the different events as they pertain to different resources. 1 unit of the resource bikes does not necessarily equal 1 unit of the resource cash. In short, we need a method to figure out whether dual events are balanced in order to report on outstanding payments for instance.

The balancing of dual events will furthermore depend on the types of these events. In a transport exchange where load events are paired in duality with unload events, we know that the two stockflows must have the same values. This is because we expect that when we send a bike to our customer, he will receive exactly one bike. In a transformation exchange, the situation is different. Here we decrement raw materials in order to receive an end-product. It is difficult to

see how the components of a bike can be balanced with the final bike resource in any general manner.

A final problem with the duality relation is the existence of *isolated events* which McCarthy already mentions in the original REA paper:

*Gains and losses are resource changes not associated with the normal earning activities of an enterprise. The exact nature of gains and losses is difficult to define and relies to a great extent upon interpretation of existing accounting practice. ... for these particular types of accounting phenomena, there are many occasions when increments and decrements occur quite legitimately in isolation.*

McCarthy [1, p. 574-5]

Intuitively, it makes sense not to try to pair an event such as theft with anything. This would be an artificial duality as there is no economic rationale in having resources stolen. Nevertheless, other isolated events do actually happen regularly and for a good reason. Paying taxes is for instance perfectly rational but we can not pair the event of paying tax with any increment [22]. Should such events be kept isolated or could we introduce a way of modeling the economic rationale behind this kind of event? This question is extremely important to software developers because it determines what kind of constraints that should be enforced in REA models. If isolated events are allowed then it is not possible to require that duality always exists and hence that economic transactions always have a rationale.

### 3.2 Temporal Properties of Events

Physical events are by nature different. Physical events for instance have varying temporal properties. The physical event of buying a bike can take less than 5 minutes. The act of producing electricity for Copenhagen, on the other hand, is a long-running activity that is only interrupted by rare blackouts. It is not explicit in the ontology how or even if we should distinguish between the differences in temporal properties.

In transfer exchanges, it makes sense to say that events are instantaneous. If we buy a bike at a store then we should be able to determine the ownership of this bike at any given moment in the process. If events had duration then there could be a period where the bike had two owners simultaneously. This should be avoided for legal reasons.

In transformation exchanges, it could be argued that events have duration. The production of electricity is a long running activity which decrements a set of raw materials and increments our electricity at a given rate. The concept of *rate* is not described in the ontology and hence it is very difficult to understand how this relates to traditional event values. Nevertheless, it does seem to represent what actually takes place in the real world that we are trying to model.

We believe that events are instantaneous by nature. The central reason is that only instantaneous events facilitate precise determination of for instance

ownership. The electricity example is not convincing because we could just as easily model the electricity production by creating events at regular intervals. These events would then be instantaneous. From the software development point of view, instantaneous events are easier to conceptualize as they are fixed, atomic entities. Events with duration would on the other hand have to be constantly observed in order to calculate the current amount of resources as well as if the production rate was stable.

### 3.3 REA Compliance

The REA ontology can be used in a prescriptive manner to constrain concrete business models. It should be possible to analyze and evaluate models that are either developed from scratch or extracted from legacy systems. In order to perform such an evaluation, we need a set of explicit evaluation criteria. We propose the concept of *REA compliance* to denote the principle by which we determine whether a concrete model conforms to the metamodel and complies with our constraints.

The REA template which is shown in figure 2 can be generalized as the main structural constraint on REA exchanges. The cardinalities of the relations in this template should be M-N as this will provide the greatest flexibility for modelers. A concrete part of these criteria could be a requirement that all events are paired in duality although this has some complications as described in section 3.1. A model complies with our criteria if all economic transactions in this model can be fitted into the general template. The repeated application of this template across large domain models at the same time gives us an understandable way of showing REA compliance and hence economic rationality.

This concept of REA compliance suggests an interesting research opportunity for people with an interest in formal methods. A formalization of the REA ontology should include a precise description of this compliance criteria. Such a formalization effort would facilitate three important contributions: First, it would be possible to use automatic model checkers to validate concrete REA models. This is necessary to enforce REA compliance in large models. Second, it would enable unambiguous descriptions of legacy models which could then be used as input for automatic validation. Finally, it would bring the vision of model-driven development (MDD) closer to realization. Formalization of REA model enables the automatic translation of models into executable code as shown in a minor case study by Borch et al. [23].

### 3.4 Modeling Compromises

The importance of implementation choices in REA-based applications is often underestimated. There are several examples of how modeling problems are ignored by categorizing them as *implementation compromises*. The issue arises when some feature of the REA ontology causes conceptual problems. A response to such lack of conceptual clarity is to exclude the given feature from the actual implementation. As previously mentioned in section 3.1, the duality relation is

hard to conceptualize hence several known implementations have simply left it out, e.g., [24, 25, 14]. Hruby et al. [14] uses the term *modeling compromise* which is probably a more appropriate term. A compromise such as leaving out duality invalidates the model according to the idea of REA compliance which we sketched in section 3.3. If a system does not have any representation of the duality relation then the compromise is on the modeling level rather than on the implementation level.

Modeling compromises are essential because they are a sign of a mismatch between the conceptual model of the domain experts and the software developers. The ontology is hardly an ubiquitous language if central concepts are not shared by both groups. Borch and Stefansen [22] states that ontology development should be a mixture of conceptual analysis, the study of real world scenarios and realistic experiments. We believe that this is a correct methodological point. It is very important that conceptual modeling is evaluated by realistic implementations. Implementations must be recognized as a valuable source of feedback rather than a mere matter of compromises. Otherwise the negotiation of the common boundary object has failed.

Legacy systems which are subjected to a REA analysis are the main exception to this rule. O'Leary [26] has for instance performed an analysis of SAP [27] which shows partial REA compliance. In this case, we are dealing with a system that has not been designed with REA in mind from the outset. The differences between REA and SAP could be interpreted as modeling compromises from a REA perspective. Such an interpretation would then provide feedback to the ontology development process and be used as inspiration for further extensions of the core ontology.

### 3.5 The Lack of Roles

The concept of roles is missing from most of the existing REA literature. In his position paper to the First International REA Technology Workshop, Hruby [28] briefly notes the lack of a distinction between agents and roles. In the UMM User Guide, a *role* relation is present but not really defined. We believe that the lack of a proper distinction between agents and roles poses a serious problem to the core ontology.

The problem is obvious in the REA models in Dunn et al.'s textbook on the REA ontology [4]. In this book, the main example of the *Robert Scott Woodwind Shop* (RSWS) provides a large paradigmatic example of a REA model. The implementation of RSWS contains tables such as *Receiving Clerks* or *Sales Representatives* which are instances of agents. Each table has columns such as *ID*, *name* and *address*. The problem arises when a person changes position from receiving clerk to sales representatives. In order to avoid data corruption, it is necessary to duplicate the information about the person that is switching job. After the switch, this person will appear in both tables and such a situation can easily lead in inconsistencies if the two records are not kept in synch.

The problem of duplication is a sign of a more general modeling compromise. There is no representation of roles in the model. The positions of receiving clerk

or sales representative are not instances of the agent entity but rather roles that an agent can play. This is really an example of how the implementation can provide feedback to the ontology development as described in section 3.4. We propose that a concept of roles should be integrated into the core ontology to solve this problem of inflexibility. A solution could be to extend the ontology by an extra entity following the *Party* archetype as sketched by Arlow and Neustadt [11] or the pattern which Hruby suggests in his position paper [28]. Another possibility would be to introduce subtypes of the participation relation corresponding to different roles.

## 4 Conclusion

The REA ontology presents a promising candidate for a grand conceptual modeling scheme for enterprise information systems. The ontology has been under development for almost 25 years now and has attracted attention from both domain experts and software developers. Despite its age, there are still several unresolved issues on the modeling level of this ontology. In this paper, we have claimed that such issues must be resolved, if the ontology is to become an ubiquitous language which can be shared by domain experts and software developers.

We have presented an interpretation of the ontology based on the central literature. This interpretation shows that the ontology is underspecified in certain areas. We have described some of these areas, viz., the balancing of duality, the temporal properties of events, REA compliance criteria, modeling compromises and the concept of roles. The discussion of these issues should provide feedback to the further developments of the ontology and hopefully bring us closer to a proper ubiquitous language.

## References

1. McCarthy, W.E.: The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment. *The Accounting Review* **LVII**(3) (1982)
2. Geerts, G., McCarthy, W.E.: The Ontological Foundation of REA Enterprise Information Systems. <http://www.msu.edu/user/mccarth4/Alabama.doc> (2000)
3. Denna, E.L., Cherrington, J.O., Andros, D.P., Hollander, A.S.: Accounting, Information Technology, and Business Solutions, 2nd. Irwin McGraw-Hill (1996)
4. Dunn, C., Cherrington, J.O., Hollander, A.: Enterprise Information Systems: A Pattern-based Approach. McGraw-Hill Publishing Co (2004)
5. Evans, E.: Domain-Driven Design - Tackling Complexity in the Heart of Software. Addison-Wesley (2004)
6. Star, S.L.: The structure of ill-structured solutions: Boundary objects and heterogeneous distributed problem solving. In: Readings in Distributed Artificial Intelligence. Morgan Kaufman, Menlo Park, CA (1989)
7. UN/CEFACT: UN/CEFACT Modeling Methodology (UMM) User Guide 5 - CEFACT/TMG/N093. Technical report, UN/CEFACT (2003) [http://www.unece.org/cefact/umm/umm\\_userguide.pdf](http://www.unece.org/cefact/umm/umm_userguide.pdf).

8. Balthazar, S., Chohan, A., Hessellund, A.: Supply Chain Integration. Master's thesis, IT University of Copenhagen, Denmark (2005) In Danish. Supervised by Kasper Østerbye.
9. Østerbye, K.: Structured REA Contracts. [www.itu.dk/people/kasper/REA2004/pospapers/KasperOsterbye.pdf](http://www.itu.dk/people/kasper/REA2004/pospapers/KasperOsterbye.pdf) (2004) Position paper on the First International REA Technology Workshop, København, Danmark, April 22-24.
10. Hessellund, A.: Supply Chain Modeling with REA. Technical Report TR-2006-80, The IT University of Copenhagen, Denmark (2006)
11. Arlow, J., Neustadt, I.: Enterprise Patterns and MDA. Addison-Wesley (2003)
12. Holmstrom, J.: Implementing Vendor-Managed Inventory the efficient way: A case study of partnership in the supply chain. *Production and Inventory Management Journal* **3**(39) (1998) 1-5
13. Jaquet, M.: REAListic - En REA-model uden perspektiver. Master's thesis, IT University of Copenhagen, Denmark (2003) In Danish. Supervised by Kasper Østerbye.
14. Pavel Hruby et al.: Model-Driven Design Using Business Patterns. Springer Verlag (2006) to be published.
15. Geerts, G., McCarthy, W.E.: An ontological analysis of the economic primitives of the extended-REA information architecture. <http://www.msu.edu/user/mccarth4/sowa.doc> (2000)
16. Geerts, G., McCarthy, W.E.: Type-Level Specifications in REA Enterprise Information Systems. <http://www.msu.edu/user/mccarth4/UTS-seminar/Type%20paper%20final%20submission.doc> (2003)
17. Sowa, J.F.: Knowledge Representation - Logical, Philosophical, and Computational Foundations. Brooks/Cole (2000)
18. Fowler, M.: Analysis Patterns: Reusable Object Models. Addison-Wesley (1996)
19. Nakamura, H., Johnson, R.E.: Adaptive Framework for the REA Accounting Model (1998) OOPSLA'98 Workshop on Business Object Design and Implementation IV.
20. Yoder, J.W., Balaguer, F., Johnson, R.: Architecture and design of adaptive object-models. *SIGPLAN Not.* **36**(12) (2001) 50-60
21. Coad, P.: Object-Oriented Patterns. *Commun. ACM* **35**(9) (1992) 152-159
22. Borch, S.E., Stefansen, C.: Evaluating the REA Enterprise Ontology from an Operational Perspective. In: *Proceedings of the CAiSE'04 Workshops*. Volume 3. (2004)
23. Borch, S.E., Jespersen, J.W., Linvald, J., Østerbye, K.: A Model Driven Architecture for REA Based Systems. In: *Proceedings of the Workshop on Model Driven Architecture: Foundations and Applications*, University of Twente, Enschede, The Netherlands (2003)
24. Gertzenstein, D.A.: An Object Oriented Framework for Business Systems Based on the REA Pattern. Master's thesis, UIUC (2003) Supervised by Ralph Johnson.
25. Wei, J.C.Y.: A REA Business Framework. Master's thesis, UIUC (2004) Supervised by Ralph Johnson.
26. O'Leary, D.E.: On the relationship between REA and SAP. *International Journal of Accounting Information Systems* (5) (2004) 65-81
27. SAP: <http://www.sap.com>.
28. Hruby, P.: Agents and roles. (2004) Position paper on the First International REA Technology Workshop, København, Danmark, April 22-24.