IT University
of Copenhagen

Proceedings of the 2nd International REA Technology Workshop

Editors:
Mette Jaquet
Anders Hessellund
Pavel Hruby
Jesper Kiehn
William E. McCarthy

IT University Technical Report Series TR-2006-92

ISSN 1600-6100 July 2006

ISSN 1600-6100

ISBN 87-7949-136-7

Copies may be obtained by contacting:

IT University of Copenhagen
Rued Langgaards Vej 7
DK – 2300 Copenhagen S
Denmark

Telephone:	+45 72 18 50 00
Telefax:	+45 72 18 50 01
Web:	www.itu.dk

Proceedings of the 2nd International REA Technology Workshop

June 25, 2006, Santorini Palace Hotel, Fira, Santorini Island, Greece

Editors:

Mette Jaquet
(mjaquet@itu.dk)
Software Development Group
IT-University of Copenhagen
Rued Langgaards Vej 7
DK-2300 Kbh. S.,
Denmark

Anders Hessellund
(hessellund@itu.dk)
Software Development Group
IT-University of Copenhagen
Rued Langgaards Vej 7
DK-2300 Kbh. S.,
Denmark

Pavel Hruby
(phruby@acm.org)
Microsoft
Frydenlunds Allé 6
DK-2950 Vedbaek,
Denmark

Jesper Kiehn
(jkiehn@microsoft.com)
Microsoft
Frydenlunds Allé 6
DK-2950 Vedbaek,
Denmark

William E. McCarthy
(mccarthy@bus.msu.edu)
Department of Accounting & Information Systems
Michigan State University
East Lansing, MI 48824
U.S.A.

Contents:

Sponsors:

IT University of Copenhagen

Microsoft Dynamics

Why accounting data models from research are not incorporated in ERP systems

P.E.A. Vandenbossche (*) and J.C. Wortmann (**)

University of Groningen

Faculty: Systems – Organizations – Management

P.O. Box 800

9700 AV Groningen

The Netherlands

Paper presented at the 2nd International REA Technology Workshop, June 25, 2006

Fira, Santorini Island, Greece

(*) Dr. P.E.A. Vandenbossche is Assistant Professor at the University of Groningen, The Netherlands

Contact: +31 50 363 38 64, + 31 6 53 65..72.79, P.E.A.Vandenbossche@rug.nl

(**) Prof. dr. ir. J.C. Wortmann is Full Professor at the University of Groningen, The Netherlands

Contact: +31 50 363 38 64, J.C.Wortmann@rug.nl

Abstract

Researchers in accounting data models proposed new accounting data models as a response to the drawbacks of double-entry bookkeeping. Two most prominent research results are: ‘Grundrechnung’ and the ‘extended REA model’. These accounting data models have not been adopted in current ERP systems. ERP systems still consider double-entry bookkeeping as the accounting data model that provides data to support many other applications.

Two questions are discussed in this paper. First, why are ‘Grundrechnung’ and the (extended) REA model not adopted in ERP data models? Second, what is the contribution of ‘Grundrechnung’ and REA model concepts for designing better ERP data models? The paper shows that some elements of of ‘Grundrechnung’ have been used implicitly in ERP data models, but that ‘Grundrechnung’ itself is not specific enough to deal with all complexities of an ERP system. Moreover, the requirement of data models being purpose-neutral has not been properly adopted in ERP. The extended REA model provides significant progress in the area of accounting data models. However, also in the current REA model specific details still need further development before REA is fully suitable as foundation for an ERP data model. These details are described in the paper.

Keywords: REA, Grundrechnung, ERP system, accounting data model

1. Introduction

Research projects in the domain of accounting data modeling started half a century ago. Double-entry bookkeeping as data source of financial information had a number of severe drawbacks. The goal was to overcome these drawbacks (McCarthy, 1980). The double-entry bookkeeping technique itself was described first by Pacioli[1] in renaissance Italy of the 15th century with the objective to record data on simple trade transactions (Geerts and McCarthy, 1997). Surviving over the next six centuries, it was afterwards also used to record data on more complex transactions, such as e.g. manufacturing value-creation processes and financial risk transactions. But double-entry bookkeeping data contains obvious limitations to support these new applications with suitable information (Vandenbossche, 2005).

Research initiatives in the domain of accounting information systems have predominantly resulted in proposals of new accounting data models, which can store objective, so-called 'application-neutral' data. These are data models that are defined independently of any application scope. They can therefore guarantee the availability of useful data to support a wide range of existing and new applications. The most prominent research results in the area of accounting data models include: principles of data recording of 'Grundrechnung' (Riebel, 1994 based on Schmalenbach, 1948, 1956) and the '(extended) REA model' (McCarthy, 1982; Geerts and McCarthy, 2000, 2002). Some authors have argued that accounting data model research can only lead to useful progress when they are the result of a combined effort between accounting and information systems design (see e.g. Davis et al., 2000). Both 'Grundrechnung' as well as the extended REA model have been defined from the accounting discipline only. It is therefore relevant to investigate whether these data models are suited in practice as foundation for accounting data models of ERP systems.

[1] Summa de Arithmetica, Geometria, Proportioni et Proportionalita (Paccioli, 1494)

ERP systems aim to be an organization's central information system over a long period of time. But organizations are not static. Both the environment in which they operate, as well as the organizations themselves evolve over time. For instance, organizations acquire other companies, they sell off divisions, move manufacturing operations to lower cost countries et cetera. As a natural result, ERP systems have to evolve also over time, in order to remain useful as central information system. New users will start to use existing information, and current users will require additional information to support new information needs. This comes down to the question whether the ERP system is capable of delivering adequate information over time in circumstances of ongoing change. The design of the ERP data environment is crucial to achieve this goal. When comparing the goals of the 'application-neutral' data model research with the user expectations of ERP systems, a clear parallel can be found in both initiatives. Both have the objective to record data in such a way that information can be provided to support a broad range of applications over a long period of time. It would therefore be of interest to ERP system architects to benefit from the progress made in accounting data modeling research by incorporating concepts of application-neutral data models in the ERP data model.

Two main questions are the subject of this paper. The first question is, why the two most prominent research initiatives in accounting data model research (i.e. 'Grundrechnung' and 'the extended REA model') have not been incorporated in ERP data models, although both types of data models pursue similar goals. The second question concerns the contribution, that 'Grundrechnung' and the extended REA model can have for the design of ERP data models. The main findings on both questions will be discussed in the remainder of this paper.

With respect to 'Grundrechnung', it turns out that some of the concepts of 'Grundrechnung' are already incorporated in the ERP data model. However, Grundrechnung' itself needs further enhancement to be useful in the complex context of an ERP system. 'Grundrechnung' introduced two main issues: firstly the principle of separation of the data environment from the application environment, and secondly the principle to store only application-neutral data in the data environment. The first principle was re-invented by ERP designers. The second principle is not yet adopted in ERP data models.

Concerning REA, the paper concludes that REA (after extensions) is designed in an object-oriented data modeling technique, which is currently not widely adopted yet in ERP data model design. Some of the REA concepts also need further enhancement to cover all required data modeling issues from a software design point of view. REA introduced the concept of coherence between data of different business events, as well as a means to define future data. In current ERP systems, the data coherence between business events is absent at the level of data model definition. This will be discussed in more detail in this paper.

The paper is structured as followed. First, in Section 2, the main components of an ERP system will be described. Next, in Section 3, it will be discussed how financial data are being recorded in current ERP systems. Afterwards, Section 4 will discuss the reasons why ‘Grundrechnung’ is being used in data models of current ERP systems to some extent. Also, the contribution of ‘Grundrechnung’ for the design of current ERP data models is discussed in Section 4. These same questions are discussed in Section 5for the extended REA model. Finally, conclusions are drawn in Section 6.

2. Components of modern business information systems

Business information systems were originally built and maintained by in-house IT departments. For cost reasons, starting in the early nineties, they were gradually being replaced by ERP systems. These are standard transaction processing information systems that can be used in many different industries by a variety of organizations. ERP systems became in the last decade of the 20th century the most important business information systems of many organizations. They are implemented to provide information services to users in various functional domains. ERP systems are nowadays surrounded by other components, such as a data warehouse, in order to represent a complete business information system. Figure 1 displays the main components of a modern business information system at high level. These main components are explained below. The three components Session Logic, Business Logic and ERP data environment constitute together the ERP system (see dotted box in figure 1).

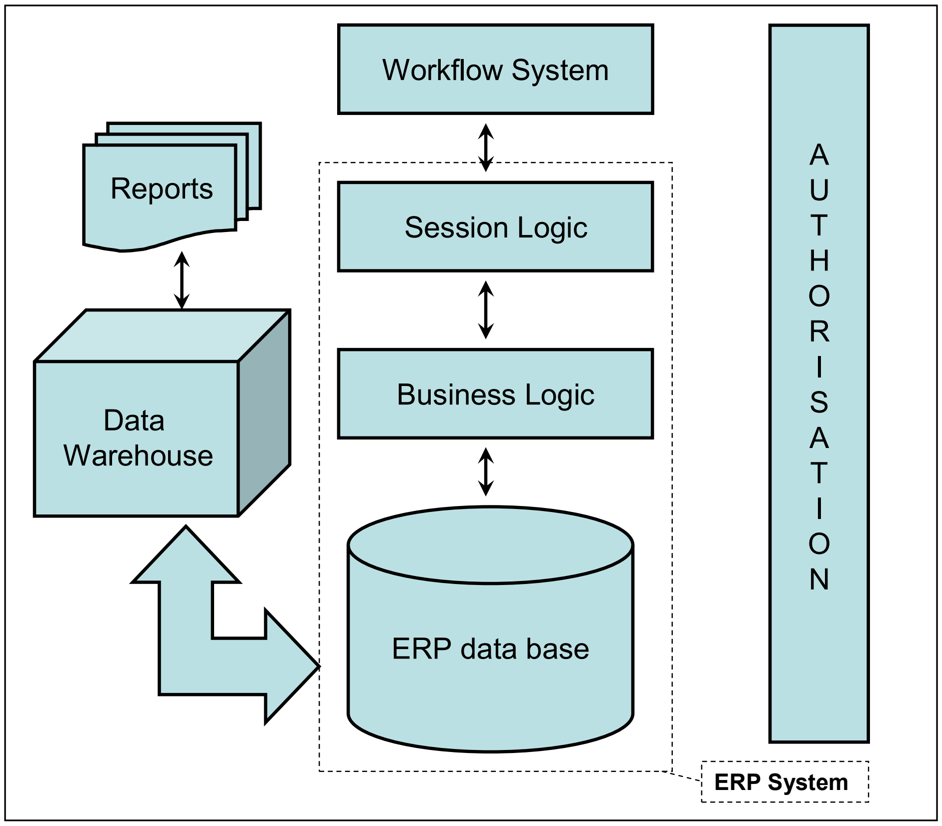


Figure 1: Main components of a Business Information System

ERP data environment. ERP systems are standard software packages, in use at many different organizations. ERP systems have a central data model, which provides required data to different functional domains in order to satisfy various information needs. ERP data models were originally defined based on analysis of these information needs in different functional domains for many potential customers. However, ERP data models have often been changed as a result of required modifications to support selected new user information needs. These data models are frozen at a certain moment in time and delivered in a subsequent ERP version. The data model of current ERP systems is therefore not the result of an application-independent data organization definition effort (guaranteeing data availability to support new information needs with current available data) but rather the derivation of originally foreseen needs, expanded by subsequent requests for enhancements. Accordingly, stored data in the ERP data environment is focused on supporting explicitly known information needs and not on supporting unforeseen needs..

Business logic. This is the application layer which converts stored data into information suitable to end users. The business logic can e.g. consist of feeding the stored data into planning algorithms to produce an optimized financial plan, or the calculation of certain tax algorithms on trade transaction data et cetera.

Session logic. This component supports the user to access the information that is produced via the business logic layer. This session logic can consist of simple data display or maintain screens, but also of a more sophisticated user interaction such as a complex graphical plan board. Sessions are primarily used as the elementary actions in transaction processing. In other words, a business transaction that has to be recorded coherently is stored in the enterprise information system via (ERP-) sessions.

Workflow systems. Sessions in ERP are used for transaction processing, i.e. to transform the ERP data environment from a consistent state into another consistent state. ERP by itself does not represent explicit knowledge on business processes, although such knowledge is generally used when designing ERP systems. Therefore, modern Enterprise Information systems often have a workflow component, where business processes can be stored and support can be given to a logical sequence of actions (i.e. a logical sequence of sessions in an ERP system).

Data warehouse. The data warehouse holds data that is derived from the ERP data environment. Its objective is to have a flexible data environment for additional information analysis and decision support purposes. As discussed earlier, the ERP data environment only holds data to support explicitly chosen user information needs. However, in practice, several additional information needs can occur on ad-hoc basis to support managerial decision making. These information needs are not readily supported by data in the ERP data environment. ERP vendors have solved this problem via providing an enrichment of the EPR data environment via data available in a data warehouse. As discussed in Wortmann and Kusters (2006), information enrichment takes place in three steps:.

In the first step, data is uploaded from the ERP data environment into the data warehouse via an ELT tool (extraction – load – transform). A data warehouse is an additional data environment, which consists of a set of tables, organized to support information needs in a specific area in a generic way. Data in the data warehouse can be generically retrieved and is not restricted by the data storage approach (e.g. table indices) of the relational ERP data environment. In this data environment, both summarized as well as detailed data can be stored.

The second step consists of transforming the data stored in the data warehouse to information required for analysis or decision-making purposes. Data warehouses often contain some build-in features like aggregation and drill-down, time-series analysis data etc. which allow end-users to convert the data into information.

The third and last step consists of presenting the data in form of a report or graphical representation. This is represented in Figure 1 in the component of 'reports'.

Authorisation. This component arranges the user access to the system and solves problems such as segregation of duties, etc.

3. Financial data in current ERP data models

The main focus of this paper is on how financial data is stored in the data model of a standard information system. The technique of 'double-entry bookkeeping' has been defined initially over six centuries ago and is commonly used by finance users all over the world as basis to support a large variety of applications with financial information. These applications were initially pure financial: accounting applications focused on information provision to external stakeholders. But over time, applications such as: manufacturing cost accounting, financial risk administration, commitment registration et cetera, were also supported with double-entry bookkeeping data. Double-entry bookkeeping is not the only financial method to provide financial information. There are a series of other financial techniques, which have been developed in the finance domain to support a particular information need (such as e.g. activity based costing, decision support techniques based on relevant cost data, et cetera). However, the use of these other techniques to provide financial information is by far overshadowed by the dominance of double-entry bookkeeping.

ERP vendors build their ERP systems based on a selection of user information requirements, which are frozen at a certain point in time. After customer information needs are being collected and frozen, they are designed, developed, and made available to end users in a next version of the ERP product (Vandenbossche, 2005). For reasons as mentioned earlier in this section, there is a very homogeneous request from all finance users of organizations in many different industries from all over the world to have primarily support of double-entry bookkeeping in information systems. This explains why all ERP vendors, independently from each other, have received a very similar request from their finance user community to support this technique in the financial information system. The way in which ERP product are being built, together with the homogeneity in request from finance users are the two most important reasons why current ERP systems all have a financial part of their data models which is designed around the concepts of double-entry bookkeeping.

Several authors in research schools of accounting data model research have pointed to the restrictions when data models are based upon application artifacts of double-entry bookkeeping such as debit / credit separators, general ledger accounts, etc. (see e.g. McCarthy, 1980; McCarthy, 1982; Everest and Weber, 1977; Vandenbossche, 2005). These restrictions relate to the fact that suitable financial data required to support future information needs are not necessarily derived from known financial methods such as double-entry bookkeeping. New financial algorithms can be proposed, which could make use of other base data. In order to overcome this dependency, the ERP data model should hold data, which is independent from any application scope. ERP data model architects have ignored this advice from research during the past few decades. The ERP data model is enhanced or modified based on features of the new information needs, which are chosen to be supported in the next ERP version. The design of the ERP data model is not an activity, which is clearly separated and independent from the application design as prescribed by research (e.g. Schmalenbach, 1956). Since double-entry bookkeeping data is being reused to support many additional financial information needs (besides financial accounting applications as discussed before), the technique of double-entry bookkeeping can be considered as the core financial application technique supported by ERP systems. Because of the dominance of the double-entry bookkeeping technique for financial end users, all application characteristics related to this technique are defined as ‘hard‘ properties in the financial part of the ERP data model. However, the financial part of the ERP data model is not only used for financial accounting purposes, but also for many other applications. Thus, double-entry bookkeeping is not just one of the many supported applications in the ERP system, but it has become the primary source of financial data and therefore a corner stone of the overall ERP data model.

One of the main reasons why research results of ‘Grundrechnung’ and the extended REA model have not been incorporated in data models of current ERP systems stems from the fact that there is no separation between the architecture of the data model and the design of the newly required application logic. In each new ERP version, enhancements and modifications are made to the data model in case additional data is required to support the chosen new information requests. In order to keep the data model changes minimal and efficient, they are limited to and based on the data characteristics of the newly supported information requests only. In the next two sections, we will investigate the contribution of ‘Grundrechnung’ and ‘the extended REA model’ to ERP data model design in more detail.

4. Contribution of ‘Grundrechnung’ to the design of the ERP system

Application-neutral data recording via ‘Grundrechnung’ as proceeding in accounting data model research has been defined by Schmalenbach (1956). The history of ‘Grundrechnung’ has been discussed in Riebel (1994), Dunn and McCarthy (1997), Verdaasdonk (2003) and Vandenbossche (2005). It started as a methodology to define data generically for a German cost accounting method (‘Einzelkosten und Deckungsbeitrags Rechnung’), in such a way that cost data are available for many costing-related decision making purposes. Schmalenbach described that cost and revenue data have to be defined independently from specific decision making situations (i.e. in an ‘application-neutral’ manner) to fit also for support of new information needs. He differentiates between ‘Grundrechnung’ (the application-neutral data environment consisting of cost and revenue base data) and ‘Sonderrechnungen’ (which are the various different applications which reuse the same cost and revenue base data as defined in ‘Grundrechnung’). Back-Hock (1995) explains the objective of ‘Grundrechnung’ as data provided to support various different information needs where no results of arbitrary allocation or valuation are stored. Independent from Schmalenbach’s ‘Grundrechnung’, Goetz (1939, 1949) argued that accountants should focus on their role as accounting information specialists instead of application specialists. From an information perspective, Goetz (1939) suggests to achieve this goal via recording a ‘Basic Historic Record’; also known as ‘the Basic Pecuniary Record’ which is a record of objective data on commitments in a business transaction (such as parties, nature of the transaction, products and services given and received, et cetera). These base data could be reused for multiple applications. Schmalenbach’s ‘Grundrechnung’ and the ‘Basic Pecuniary Record’ of Goetz pursue similar goals (Schweitzer, 1992). Both authors promote the concept of a data environment where primitive, objective data are stored, uninfluenced by the specific information needs of certain information users. They describe concepts of application neutral data registration several decades before computer-based information systems were commonly used in organizations. It took long before subsequent research towards improved accounting data models has been conducted, which startedg in the early eighties, e.g. initial versions of the REA model (McCarthy, 1982). Riebel (1994) has further operationalized Schmalenbach’s concepts of ‘Grundrechnung’. He proposed generic data recording principles, which encompass the original goals of German cost accounting purposes in the early nineties, at a moment where legacy information systems were commonly being replaced by standard information systems as ERP systems. He outlined the following data recording principles (Riebel, 1994):

- *Data recording principle #1: No heterogeneous classification or summarizing of elements needed separately for applications*
- *Data recording principle #2: No arbitrary division and allocation of accounting data*
- *Data recording principle #3: Entries are to be recorded at the lowest level possible in the hierarchy, without introducing arbitrary allocations*
- *Data recording principle #4: Only characteristics with all attributes of interest and importance have to be recorded*

The data recording principles of 'Grundrechnung' can be considered as guidelines to follow when designing an application-neutral data environment focusing on providing information for various applications. Since the goals of ERP data models are similar, it is relevant to investigate their value to designing an ERP data model.

In the remainder of this section, we will discuss the contribution of 'Grundrechnung' for designing ERP data models via a discussion on the four data recording principles. Four findings will be presented. These findings will point out, that whilst ERP data models do not fully comply will all 'Grundrechnung' data recording principles as originally pursued, current ERP data models have solved the problems discussed by 'Grundrechnung' in a different way (e.g. via adding a data warehouse).

Data recording principle #1: No heterogeneous classification or summarizing of elements needed separately for applications.

This data storage principle was also pursued by Goetz (1949) who pursued recording of business transactions in their original form so that it can be reused and enhanced to fulfill the information needs of a broad variety of information users. This recommendation was new and important even a decade later when IT was in its infancy and hardware capacity was still expensive. In that timeframe, most business information systems, if automated at all, were proprietary information systems which were built and maintained by in-house IT departments. Their data models were designed as efficient as possible in order to minimize the required data storage capacity. As a logical consequence, this approach resulted in propriety data models which stored on purpose predominantly heterogeneous and/or summarized information in one and the same data model, so to obtain efficiency gains. These data models were specifically designed to only fulfill the information needs of specific users adequately at one point in time. But with the introduction of data base management systems in the '60s and relational databases in the '70, homogeneous data are commonly stored in the databases.

The first ERP systems were built in the early nineties based on modern database management systems in a context where database capacity was not expensive anymore. The problem of storing heterogeneous data is therefore now outdated and long solved due to the availability of modern and inexpensive database management systems. Summarized data is not stored in the ERP data model. When needed, such data is stored in an auxiliary data warehouse which holds derived data from the ERP data model as visualized in Figure 1. The functionality of a data warehouse allows to generate summarized or aggregated data and keeps it persistent in the data warehouse. Calculated values can be re-generated at any point in time and are therefore in principle redundant.
Assessment: The issues of heterogeneous and / or summarized classifications are long solved by modern data base management systems. In current ERP systems, summarized data is stored in an auxiliary data warehouse management system separate from the ERP data model if convenient for analysis or reporting purposes.

Data recording principle #2: No arbitrary division and allocation of accounting data.
'Grundrechnung' was originally defined to provide data for a German cost accounting application, where the objective was to store data independently from application schemas in order to guarantee its reusability. This data recording principle refers back to a generalization of the original problem solved by 'Grundrechnung', i.e. the requirement to store application-independent, primitive base data separated from any chosen decision making model as introduced by Schmalenbach (1956) and Goetz (1949). But how does the essential benefits of a purpose neutral data models translate to current information systems? The separation between the data model and the application environment as outlined by Schmalenbach and Goetz are well-known design patterns in modern business information systems, such as ERP systems or CRM systems. The advantages of a purpose neutral data model relate to more fundamental functional design choices. According to this concept, data is being defined and structured 'objectively' without its definition is being impacted by characteristics of any application scope. A consequence of this approach of data structuring should be, that this would not lead to restrictions when data has to be reused to support new information needs in the future. This latter concept has not been adopted in architectures of current ERP systems. As explained in Section 2, modern ERP systems have

automated the double entry bookkeeping systems as a result of the fact that this was the most frequently recurring user request from the finance user community. The fact that today's database management systems allow to separate the data model from the application logic as a natural design choice has incorrectly been interpreted by information system architects as the achievement of a purpose neutral data model. Schmalenbach (1956) has not provided an adequate explanation via 'Grundrechnung' on how ERP data model architects have to structure a purpose neutral data model in practice. In the next section, it will be explained why McCarthy's extended REA model has provided a better solution to this problem

Assessment: The architecture of ERP systems recognizes the separation between the data model and the application model. This separation is normal in modern business information systems but is does not result in a purpose neutral data model as set forward by researchers in the domain of accounting data model research.

Data recording principle #3: Entries are to be recorded at the lowest level possible in the hierarchy, without introducing arbitrary allocations.

In general, ERP systems do record data at the lowest possible level. The lowest-level data relate to e.g.: individual purchase orders with purchase order line detail, individual purchase invoices with purchase invoice line detail, detailed payment document registration, et cetera. In some circumstances, ERP systems offer the possibility to either store data at the lowest level or to store data after grouping at a more aggregated level (e.g. calculation of minority interest in a consolidation cycle). The latter data recordings do not relate to storage of primitive base data, they are supported by the ERP system via a secondary data environment for optimization and efficiency reasons. For instance, whilst primitive base data on purchase orders is always stored at purchase order line level detail, it can be an implementation choice to post the purchase order data to the general ledger with various levels of detail. Examples could include e.g.: full detail of purchase order lines is posted to the general ledger via individual journal entries; only one journal entry for the entire purchase order (i.e. aggregation of all lines) is posted to the general ledger; or even only one journal entry holding the aggregated data of all purchase orders of one business partner is posted to the general ledger. In these examples, it should be reemphasized that these are all secondary data recordings made for efficiency and optimization of the information provision. These do not alter the lowest level base data stored in the ERP data model, which was in this case: data recording of purchase orders at purchase order line level.

Assessment: Base data at the lowest level without allocations is always recorded in ERP data models. There are no exceptions to this. ERP systems sometimes maintain a secondary data environment for optimization of data retrieval an information provision. In that environment, it is legitimate to only store aggregated data to achieve the goals of optimization. This additional data storage does not change the primitive base data.

Data recording principle #4: Characteristics with all attributes of interest and importance.
Riebel's (1994) recommendation on data storage with all attributes of interest and importance has to be understood from an application-neutral, objective perspective. This means that only relevant, recurring and objective primitive data elements may be stored in the base data environment. ERP systems store data on business transactions in the ERP data environment. As explained earlier in this section, the characteristics of the data environment are not purpose neutral but influenced by the nature of information needs supported. Contrary to this 4th data recording principle, stored attributes of interest and importance in ERP systems are dependent on the chosen application scope. This approach limits the possibility to support other information needs for which other attributes of interest are important. ERP systems have provided some flexibility by adding a data warehouse to the ERP data environment. A subsection on the required data is uploaded from the ERP data base in to the data warehouse. These data warehouses are supported by OLAP (on-line analytical processing) tools, which allow to run data inquiries from all possible perspectives. The attributes of interest to new information needs that originally were not focused on in the ERP data environment, can be brought together. This allows to some extent more information needs to be supported via data currently available in the ERP data model.
Assessment: 'Attributes of interest and importance' vary by data requirements to support different information needs. The original objective of 'attributes of interest and importance' irrespective of an application scope has not been followed by ERP systems. But the addition of data warehouse techniques to enrich the data environment provides some options to reach the same goals.

Conclusion on the contribution of 'Grundrechnung' for ERP data models:
The ideas of 'Grundrechnung' were defined the middle of the previous century at a moment in time when manual information systems were not commonly replaced by IT information systems. There was no need for efficient and future-oriented data organization via data models yet, since IT information system design was still its early days. ERP systems do not fully comply with all concepts of 'Grundrechnung' as originally prescribed by Schmalenbach (1956), Goetz (1939, 1956) or even Riebel (1994), but they attain several of the goals as set forward by 'Grundrechnung'. The base concept of separation of the data environment from the application

environment has been followed in ERP systems, but data stored in the data environment of ERP systems deals with application driven business transaction data which is not application-neutral as prescribed by ‘Grundrechnung’. The base data stored in ERP systems does comply with all data recording rules as outlined by Riebel (1994). The fundamental restriction of ‘Grundrechnung’ to be fully useful in ERP data model design, lies in the fact that most of the problems solved by ‘Grundrechnung’ are now outdated and nowadays considered as commonly chosen standard design choices. There is no real guideline in ‘Grundrechnung’ on how to organize an application-neutral data model for a complex business information system as an ERP system. In fact, the best guideline can be found by inspecting the REA-model, which can be said to propose a really ‘purpose-neutral’ data model as will become clear below.

5. Contribution of ‘(extended) REA model’ to the design of the ERP system

The REA model is the first semantic, application-independent data model, which has been defined in the early eighties when the drawbacks of double-entry bookkeeping as data source became well understood. The first REA model was defined around the key components: Resource – Event – Agent, and various relevant relationships between them (McCarthy, 1982). Originally, the REA data model was designed following the entity-relationship design methodology (Chen, 1976), which was commonly used at the moment when the REA model was initially defined. In this E-R methodology, real world phenomena such as invoices or contracts are modeled as entity types. Entity types can have relationships. For examples, contracts have relationships with their clauses, invoices have relationships with clauses, etc. Sakagami (1995) pointed out the drawback inherent to the fact that the REA model was designed via the E-R methodology. The consequence of using this design method meant that the data model could face restrictions when deployed in real-life customer implementations since the number of REA components would grow exponentially. Geerts (1997) described this problem as a lack of ‘reusability’ and ‘extendiblity’. This drawback was recognized and later an object-oriented version of the REA model was proposed (see Dunn and McCarthy, 1997; Geerts and McCarthy, 1997; and McCarthy, 1995). This data modeling methodology allows to incorporate structural and behavioral aspects of business objects in the data model. Examples of such business objects are again e.g. contract, order, invoice, payment, etc. Structural aspects refer to the internal structure of business objects. Thus, a contract is modeled including its clauses, an order is modeled including a list of order lines, and an invoice is modeled with all details leading to the bottom line. The effect of modeling activities is that business objects are represented in information systems according to the models specified. Behavioral aspects of business objects refer to the behavior of modeled business objects in the information system. More specifically, business objects in information systems publish methods, which can be used by software for enquiring their content and form. An important feature of the object-oriented methodology is inheritance: the possibility to *inherit* structural and behavioral aspects of business objects. This feature takes away most of the drawbacks of REA as initially formulated using the E-R methodology.

The first REA model only focused on supporting a series of financial accounting information needs, which were all requiring historic data only. The REA model has been further developed into the extended REA model and a REA domain ontology (Geerts and McCarthy, 2000, 2002). The REA domain ontology overcomes much of the original REA model's scope limitations. Just as in case of the original version of the REA model, an object-oriented variant was designed for the extended REA model, by Hruby (2004) and is visualized in Figure 2. The contribution of the extended REA model for ERP systems will be discussed based on this variant of the REA model.

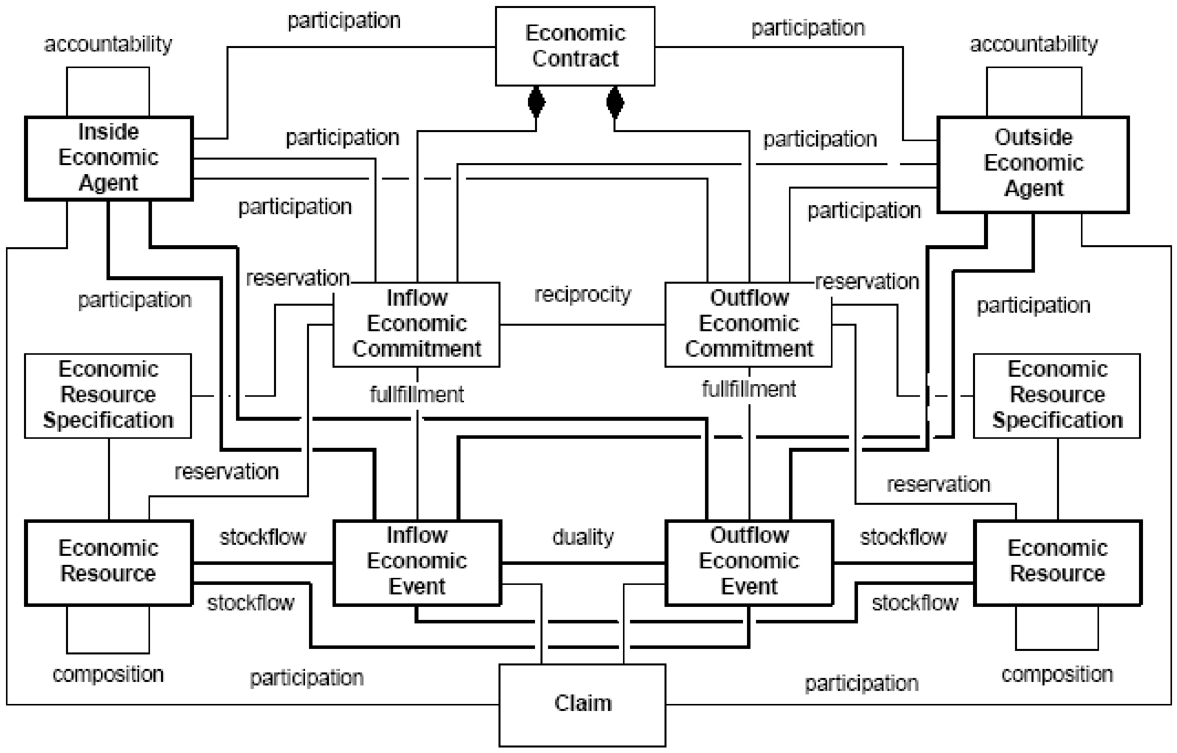


Figure 2: Object version of the extended REA Model (Source: Hruby, 2004)

In the remaining part of this Section, 5 findings will be discussed on the contribution of the extended REA model for ERP data modeling. Finding #1 will explain why an object data model (in general) is not really suited as enhancement for current ERP data models, because ERP systems are not designed following an object orientation design method. Finding #2 will provide reasons why the REA concepts to define future data are not ready for adoption in current ERP data models. Finding #3 will discuss the contribution for ERP data models of REA concepts to define the coherence between data of business events of one or more business processes. Finding #4 will discuss some conceptual gaps as well as the lack of detail of extended REA to cover all complexities supported by current ERP systems. Finding #5 explains the contribution of the extended REA model to define business process information explicitly in the same framework as business event data.

1. *The extended REA model is proposed as object data model but current ERP data models are designed following more traditional design methods*

Murthy and Wiggins (1993) discussed in the early nineties that object data models are the most interesting area for research towards improved accounting data models. In line with the advice, the first REA model was initially converted from an entity-relationship model towards an object data model (discussed in various publications as mentioned earlier in this section). The extended REA model was afterwards also proposed as object data model (Hruby, 2004). In order to be useful as extension to or replacement of current ERP data models, this would require that current ERP data models are also defined as object data models. But the largest ERP systems currently being sold and deployed do not use object-oriented development languages or object databases. They are built in traditional 3GL or propriety 4GL-based development languages[2] for which the entity-relationship design method is still being used today to enhance the system. Unless a decision would be made to build a new ERP system using modern object-oriented development languages and object database technology, proposed accounting data models in object-orientation (as e.g. the extended REA model) will not easily be adopted in practice in current ERP systems.

Assessment: Recent research results in accounting data modeling research (like the extended REA model) are proposed in the newest data modeling techniques (i.e. object-orientation). Only in the

[2] Software development languages of the 4 largest ERP systems today: SAP R/3 built in propriety 'ABAB' (new mySAP built in object-oriented J2EE); PeopleSoft (Oracle) built in propriety 'PeopleCode'; SSA ERP LN built in propriety 'SSA/Baan 4GL'; Lawson built in Cobol.

last few years, the first commercially viable software products were built in object-oriented software languages designed via this technique. The large ERP systems currently deployed are built in traditional 3GL or propriety 4GL-languages and designed based on the entity-relationship technique. This explains why (extended) REA model concepts are not found in current ERP systems.

2. The extended REA model is a detailed example of an application-neutral data model. It provides concepts to store future and past data consistently. These are useful concepts in case of new ERP data model design, but cannot be implemented as extension of existing ERP data models

The first REA model (McCarthy, 1982) was focused on supporting financial accounting applications and could therefore be considered as a data source alternative to the general ledger. Only historic information was provided. The extended REA model (Geerts and McCarthy, 2000, 2002) has added the concepts ‘the economic agreement’, which can consist of ‘economic contracts’ and ‘economic schedules’ to capture future data. The latter two concepts deal with future exchange resp. reservation of resources and are based on the same REA concepts that were already in place to detail historical data. The contribution of the extended REA model in this context is that one application-neutral concept (i.e. REA components including required relationships) is used to support both future as well as historic data. Unlike ‘Grundrechnung’, the extended REA model is a quite detailed and elaborated proposal of an application-neutral data model. Explicit semantic data components are being proposed together with the required inter-relationships, whilst excluding all possible influence of application characteristics. In other words, the extended REA model is an application-neutral data model, which maintains a clear separation of the data environment and the application environment (idea originally described by Goetz (1939) and Schmalenbach (1956)). Could some or all of these concepts be reused in current ERP data models?

Current ERP data model are an automation of the double-entry bookkeeping system. The data model is defined around general ledger accounts and concepts of double-entry bookkeeping. They focus on providing historic data as a result of user requirement preferences. In some cases, budgeting on general ledger accounts is supported as well, which can be seen as a limited provision of future data. Reusable future financial data is not available otherwise. Question is whether the REA concepts to define future data could be used as an extension of current ERP data models to record more comprehensive information. In order to consider the REA concept as useful and transparent concept for ERP data models to record both future as well as historic data, this would mean that the general ledger concept has to be replaced by REA first. This could be a feasible option when building a new ERP system. But in current ERP system, double-entry bookkeeping became the data model rather than one of the many supported applications. Hence, incorporating concepts and features of REA are directly conflicting with data definitions of double-entry bookkeeping.

Assessment: As long as double-entry bookkeeping is the basis of ERP data models, and ERP data models are not defined following the design philosophy of application-neutral databases, no concepts of the extended REA model can be reused or adopted in the ERP data model. Only when a purpose neutral data environment is the data fundament of an ERP system, ongoing enrichment and reuse of data stored in this environment can be realized via reusing the concepts of (amongst other data models) the extended REA model.

3. *The extended REA model provides concepts and relationships to define business event data in coherence at data level. These concepts are useful to build new data models, but cannot be easily implemented as extension of current ERP data models*

Different business events often have relationships, which cannot be represented by classical accounting data models (notably double-entry bookkeeping). For example, a contract may be related to a shipment and the shipment may be related to an invoice, without explicit representation in the financial data model. The term 'in coherence' is used here to denote explicit representation of such relations in data models. The extended REA model has recognized the importance of recording data on the coherence between business transaction data of a single business event as well as between different business events. Examples of concepts providing 'coherence'include 'the REA Value chain' (specifying related business processes); 'the REA Process' (defining related REA events) (see Geerts and McCarthy, 2000, 2002). Future information needs may require any combination of available data. Hence, the relationships between data of one or more business processes have to be available in the accounting data model, independent from applications. In current ERP systems, coherence is sometimes provided by data models in other domains than accounting. Even then, specific application logic is required to maintain the coherence between data of one or more business events since no coherence is maintained at accounting data model level. This a serious restriction to the reuse of available data to support future information needs. The various relationships and concepts provided by the extended REA model allowing to recording data coherence in the data model itself are therefore a significant contribution for current accounting data models. However, the REA concepts on data coherence cannot be easily adopted in an existing ERP data model since they interfere with the data coherence, which supported via ERP application software logic and data models in other areas. The data relationships maintained via application logic should be removed first, before relationships at data level could be implemented. This would incur a drastic rewrite of a large part of the ERP software code.

Assessment: The extended REA model records information on the coherence between data of one or more business events, which is a significant contribution compared to current accounting data models. In current ERP systems, these relationships are maintained in the application logic and in data models in other domains. Implementation of concepts of the extended REA model to support data coherence at data level in current ERP systems would imply a large rewrite of software code to remove specific relationships defined in the application logic, and is therefore not a realistic option for existing ERP systems. In case one decides to design a new ERP system, the extended REA model could be considered. However, the danger of redundancy with data models in other enterprise domains remains a significant challenge. This point wil be elaborated in the 4th finding below.

4. *The extended REA model still lacks specific detail and certain concepts to record all data used in an ERP system in an explicit way.*

The scientific publications on the REA model describe the data model concepts at high level without explicit detail (e.g. McCarthy, 1982; Geerts and McCarthy, 2000, 2002). Recently, REA conferences were held (e.g. the First International REA Technology Workshop April 22-24, 2004, Copenhagen, Denmark as well as the Second International REA Technology Workshop, June 25th, Santorini, Greece) where authors proposed specific extensions to the REA model. Most of this progress on REA thinking is still in the stage of unpublished conference papers, some progress even in local language. In this paper, the objective is not to propose extensions to the extended REA model to make the REA model a fully suitable data model for an ERP system. Only some areas in the REA model will be highlighted where additional detail would be helpful for the adoption in ERP systems.

A central objective of ERP systems is supporting business processes on reservation or exchange of resources, followed by the fulfillment (i.e. the execution of the reservation or exchange). The first two concepts (reservation and exchange) are explicitly supported by the extended REA model. The latter concept (fulfillment) is currently supported in the extended REA model via the concept of ‘events’. The fulfillment of a resource exchange describes the details of aspects such as:

- how were goods delivered or received?
- for what amount were the goods or services invoiced?
- how was the payment agreed to be done?,
- was there a (partial) refusal of the goods – and has a credit note been issued for this ? etc.

Apart from providing the concept of 'events' at high-level, the extended REA model is not capturing the detailed information on this part of the business process. As soon as the REA accounting model extensions will cover these concepts, overlap and redundancy with data models in other domains will emerge.

The REA model has been defined based on aspects of reality instead of application artifacts so to avoid possible restrictions in the data definition (McCarthy, 1982). Whilst the (extended) REA data model does not hold application-based restrictions, question is whether this approach indeed leads to a data model that holds sufficiently detailed data, representative to support information needs of current and future ERP users. The data in an ERP system is based on "entity types". Examples of these entity types include: 'the sales order', 'the sales quotation', 'the delivery note', 'the payment document', 'the credit note', et cetera. Entity types are often equated with the 'forms' in which they are displayed ('order-form', 'quotation form', 'delivery form', etc.). The extended REA model only holds data on the REA components and relevant relationships of each business event, but perceives the presentation of these data on a 'form' already as an application artifact. Whilst this approach could be defended from a principal and theoretical viewpoint, question is whether or not the chosen REA approach indeed holds sufficient data. Alternatively, one could argue that the entity type ('form') should also be considered as an example of an aspect of reality in the context of ERP systems, since this is an inherent characteristic of how business data are recorded in reality. Following the latter perspective, one would expect data components similar to ERP entity types in the extended REA data model. More specifically, the extended REA model will require business objects such as order, delivery, invoice and payment, all with the complexity found in ERP packages. In business reality, there also exist purely administrative steps like an 'invoice', a 'credit note' which are only a derivative of the processes of resource exchange or reservation currently stored in the REA model. Since amounts on an invoice or a delivery note or a credit note could be significantly different from the original agreed upon resource exchange (e.g. invoicing in 4 installments), additional business objects (like e.g. ERP 'entity types') have to be provided in the REA data model in order to accommodate all required information for an ERP system. However, such extension of the REA accounting data model will lead to additional overlap with existing ERP data models and functionality in non-financial domains, and therefore more (interdisciplinary) research is needed to create working solutions.

Some more detailed comments are appropriate here. The extended REA model is defined at a level of abstraction where some concepts are implicitly assumed. For instance, the REA concept of 'duality relationship' (McCarthy, 1982) outlines how an increment in a certain resource corresponds with a decrement in another resource in case of a resource exchange (McCarthy, 1982). But no information is explicitly stored on terms and conditions under which a resource exchange takes place, e.g. payment terms, delivery terms, et cetera. These are implicitly assumed to be part of the mentioned relationship. Similar examples are e.g. the (REA) Economic Event (representing an obligation with various degrees of enforceability on actual consumption or acquisition transactions), the (REA) Commitment (detailing the obligation on a reservation of resources). Both concepts assume the detail on what obligations are being defined, but these are not explicitly part of the REA model. A large part of the optimization via an ERP system is based upon this level of detail (e.g. the automatic sales invoicing is generated based upon the payment terms, the automatic delivery of goods is triggered by delivery terms). Also, a significant number of information request related to performance indications deals with this information (e.g. which customers have been paying according to payment terms, have we delivered our customers in time, etc.). For these reasons, a number of REA concepts currently assumed implicitly in the REA model should be made more explicit.
Assessment: The REA model was defined at high level and requires an additional level of semantic detail in order to become sufficiently complete to be suited as data model for an ERP system. Some concepts are currently hidden in REA relationships, and could be made more explicit (e.g. terms and conditions). Some other concepts should be added (e.g. concepts to store specific detail on various types of fulfillments).

5. The extended REA model provides concepts to define business processes explicitly in the same framework as business event data
In current ERP systems, the concept of a business process is not supported explicitly. The application software code supports the behavior of various individual business process steps, but there is no central definition of the business process stored in the ERP data model. The initial REA model (McCarthy, 1982) only focused on individual business events without recognizing the fact that these business events are part of a business process. Data was recorded on individual business events, very similar to the way of data recording in ERP systems (where relations

between individual business events are not defined at data level but via application logic). The extended REA model has introduced the concepts of the 'business process' and the 'value chain' to group individual REA instances (McCarthy, 2003). In order to organize the detail of a REA 'business process', an additional concept (the 'REA task') was introduced to decompose business processes into different business process steps (Geerts and McCarthy, 2000). In this way, the extended REA model emerged, in which both the business process (including individual business process steps) as well as the data of the business process steps could be expressed. This is an important contribution to support the complexity dealt with in current ERP systems. Current ERP systems support the full business process layer implicitly in the application logic.

As shown in figure 1, external work flow management tools can be integrated to an ERP system. Via a work flow management tool, the business process definition can be made more explicit, however the business logic to execute the business process remains hidden in the ERP software code. Therefore, there is a risk that redundancy is created between problems solved in the workflow context and problems solved in ERP. This may lead to difficult implementation problems when combining ERP and work flow systems (See Szirbik and Wortmann (2005)). Essentially, architectural guidelines are needed

The extended REA model provides a definition framework for both the business process and the data. For ERP systems, the REA business process architecture could be considered as an alternative to the use of workflow management systems. However, this will lead to the same problems as currently encountered when combining work flow and ERP. When it comes down to the level of 'REA task definitions', a major conflict may exist between the definition of the REA task on one hand, and the definition of the ERP business process step (the ERP "session"), which is defined in application software code.

Assessment: The extended REA model proposes a framework to define both the data and the business process definition in one and the same framework. Whilst this is a first step to guarantee consistency and completeness of the system, a strict separation of data and application logic (as pursued by 'Grundrechnung', see Schmalenbach, 1956) remains important. Since different concepts are used in the extended REA model for data and business process definition, the required separation is well maintained. The proposed REA business process definition model is probably only suited for adoption in newly to be build ERP systems.

Conclusion on the contribution of 'REA' for ERP data models:

ERP data models are based upon double-entry bookkeeping paradigms. The extended REA model is an alternative data model especially designed to overcome the drawbacks of this technique. Hence co-existence of concepts of current ERP data models and the extended REA model principally out rule each other in the same data environment. For this reason, complementary concepts offered by the extended REA model cannot be implemented as extension of current ERP data models. In case a new ERP data model was to be designed, the extended REA model could be chosen as approach. The extended REA model has conceptually solved a number of complexities perceived in current ERP data model definition. Most important improvements relate to definition on coherence of data of business transactions of one or more business processes as well as an application-neutral data organization concept to store historic as well as future data. However, some of the REA concepts are either to implicit or should be described with some more detail in order to hold all required data used in an ERP system.

6. Summary and Conclusion

ERP systems have similar goals as application-neutral accounting data models defined in research, such as ‘Grundrechnung’ and the ‘extended REA model’. Hence it would be interesting for ERP systems to reuse these research results when designing a new ERP data model. This, however, has not been the case in the past 15 year for any of the ERP systems. Two questions constitute the subject of research in this paper. First, it is investigated whether both data models solve problems currently being unsolved in ERP data models. Second, the reasons why these research initiatives were not adopted in current ERP data models are investigated. Both research questions are now revisited and some recommendations for the future research are made.

The most substantial contribution of ‘Grundrechnung’ is the strict separation of the data area from the application area, as well as the approach of formulating data recording principles (in general) against which any new data model could be validated. However, ‘Grundrechnung’ was initially defined in the middle of the previous century (prior to massive use of IT information systems) with the first focus to solve the required flexibility for a German cost accounting application. Given this situation, current ERP data structures today do comply with the ‘Grundrechnung’ data recording principles as currently defined. The idea of proposing data recording principles is of significant value when designing an ERP data model. However, in order to offer a significant contribution, it is recommended to define a new set of data storage rules, which explicitly focus on topics currently unsolved in ERP data models. Examples include e.g.: the definition of coherence between data of one or more business processes, a coherent definition of future data with historic data, et cetera. In order to be useful to the design of ERP data models, it is crucially important that data recording principles are defined with a significant level of detail and its use and best practice illustrated via a sample data model.

The contribution of concepts of the extended REA model to the design of ERP data models deals with the fact that a conceptual solution is proposed to several aspects which are currently either not solved- or solved in ERP systems in a restrictive way. Examples include e.g.: a single framework to define business process information as well as business event data, a single concept to define historic as well as future data, definition of coherence between data of one or more business processes et cetera. The reasons why none of these concepts has been adopted in current ERP data models relate to the fact that the REA concepts and the current ERP data models (based on general ledger account information) cannot be integrated without introducing data conflicts. As well, the REA model is expressed as object data model whereas current ERP data models are designed via traditional E-R data modeling design techniques. Another reason relates to the fact that some additional detail and some required concepts are missing. For all these reasons, the extended REA model as currently proposed only seems useful to design new ERP data models, a situation which will hardly occur in current mature and consolidating ERP software markets. Hence, we recommend to search for possibilities to modify or enhance the REA model in such a way that it can either coexist with existing ERP data models or, even better, that some concepts of the extended REA model could be redefined as extension of current ERP data models. Redefining the extended REA model in an E-R design method seems a logic and required step to achieve this goal.

The extended REA model and data recording principles of ‘Grundrechnung’ have been defined to solve a problem of accounting data definition without involvement of IT science or practice. Perhaps this is the most fundamental reason why no adoption of concepts of these data models has been taking place in current ERP data models. In order to introduce research results of accounting data model research in ERP data models, future research initiatives should have a highly involvement of IT science and practice, and focus to propose solutions which are an improvement of current ERP data models without replacing these data models.

Bibliography

Back-Hock, A. (1995). A structuring of the database and methods for a workflow accounting information system. *AIS Research Symposium,* Phoenix, AZ (February 2-4).

Chen, P.P. (1976). The entity-relationship model – towards a unified view of data. *ACM Transactions on Database Systems*, March, pp. 9-36.

Davis, J.S., Gerard, G. and W.E. McCarthy (2000). Design Science: building the future for AIS.

Dunn C.L. and W.E. MCarthy (1997). The REA Accounting Model: Intellectual Herritage and Prospects for Progress. *Journal of Information Systems*, 11 (1), pp.31-51.

Everest, G.C. and R. Weber (1977). A relational approach to Accounting Models. *The Accounting Review*, April, pp. 340-359.

Geerts, G.L. (1997). *The Timeless Way of Building Accounting Information Systems: the 'Activity' Pattern.* Paper presented at the OOPSLA 97 conference. Atlanta, Georgia.

Geerts, G.L. and W.E. McCarthy (1997). *Using Object Templates for REA Accounting Model to Engineer Business Processes and Tasks.* Paper presented at the 20e Annual Congress of the European Accounting Association, Graz, Austria.

Geerts, G.L. and W.E. McCarthy (2000). *The Ontological Foundation of the REA Enterprise Information System.* Paper presented at the American Accounting Asssociation Conference. Florida 2000.

Geerts, G.L. and W.E. McCarthy (2002). An ontological analysis of the economic primitive of the extended REA enterprise information architecture, *International Journal of Accounting Information Systems,* 3, pp. 1-16.

Goetz, B.E. (1939). What's wrong with accounting. *Advanced Management (Fall)*, pp. 151-157

Goetz, B.E. (1949). *Management planning and Control*. New York, McGraw-Hill.

Hruby, P. (2004). Several Business Patterns. Draft paper for submission to VikingPLoP 2004, Denmark.

McCarthy, W.E. (1980). *Construction and Use of Integrated Accounting Systems with Entity-Relationship Modeling*. in P. Chen, ed. *Entity-Relationship Approach to Systems Analysis and Design* (North Holland Publishing Company, 1980), pp. 625-637.

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

McCarthy, W.E. (1995). *The REA accounting model framework.* Paper presented at the OOPSLA 95 conference, Austin, Texas.

McCarthy, W.E. (2003). The REA Modeling Approach to Teaching Accounting information Systems, Issues in Accounting Education, Vol. 18, no. 4, November 2003.

Murthy, U.S. and C.E. Wiggins (1993). Object-oriented modeling approaches for designing accounting information systems. *Journal of Information Systems*, 7 (2), pp. 97-111.

Riebel, P. (1994). Core Features of the ‘Einzelkosten- und Deckungsbeitragsrechnung’. *The Accounting Review*, 3 (3), pp. 515-543.

Sakagami, M. (1995). Introducing an object-oriented model into accounting. *Osaka City University Business Review*, 6, pp. 11-25.

Schmalenbach, E. (1948). *Pretiale Wirtschaftlenkung, Bd 2: Pretiale Lenkung des Betriebes*. Bremen-Horn [etc.]: Industrie- und Handelsverlag [‘Prices in a planned economy, Part 2: Price policy in organizations’ – in German].

Schmalenbach, E. (1956). *Kostenrechnung und Preizepolitik*, Cologne, Opladen, 7th ed.

Schweitzer, M. (1992). Eugen Schmalenbach as the founder of cost accounting in the Germany-speaking world. *Collected papers of the Sixth World Congress of Accounting Historians, Kyoto, Japan.* Volume II, pp. 292-418.

Szirbik, N.B. and J.C. Wortmann (2005). ERP and workflow. *Advances in Production Management Systems,* Arlington MD.

Vandenbossche, P. (2005). Accounting information for changing business needs – Concepts of business logistics applied to treasury management decisions. PhD. Dissertation University of Groningen, p. 209.

Verdaasdonk, P.J.A. (2003). An object-oriented model for ex-ante accounting information. *Journal of information systems* 17 (1), pp. 43-61.

Wortmann en Kusters (2006). *Enterprise Models and Enterprise Information Systems.* To appear, published by Elsevier.

A Property Driven Approach towards Describing Semantics of REA Entities

Presented at The 2nd International REA Technology Workshop 2006, Santorini

Mette Jaquet
Software Development Group
IT-University of Copenhagen
mjaquet@itu.dk

June 25th, 2006

Abstract

The Resource Event Agent (REA) ontology as it has been presented so far does not explicitly describe the semantics of economic events. The lack of well-defined semantics for REA entities requires implementation experts to perform subsequent analysis of specific solutions based on models created by business experts, thereby leading to individual interpretations and decreased reusability, adaptability and evolvability of solutions. This paper presents a hypothesis that the lack of well-defined semantics for events is closely related to the lack of descriptions of properties of resources, and that choosing properties of resources as the starting point for REA modeling guidelines, can improve flexibility, adaptability and evolvability of both models and implementations of REA based systems. Using such techniques to enhance REA models could improve the consistency of models as well as shorten the path from business model to operational implementation of enterprise systems, and improve the possibilities of reusing domain knowledge of specialized semantics in different business areas.

Introduction

It has been a quarter of a decade since the REA model was first presented by William E. McCarthy as a powerful and innovative approach to modeling ERP systems. In the early descriptions of the REA model [McC79], [McC82] it was tailored to fit the context of accounting requirements in a single retail enterprise, and the extended REA ontology presented by McCarthy and Geerts in 1999/2000 [GM00] is still strongly influenced by that scope. If the context is extended to represent more than one enterprise, i.e. in when modeling Business to Business transactions from a global perspective or Supply Chain Management scenarios like Vendor Managed Inventory, this enterprise specific or view-dependent version of the model is not appropriate. Another extension to the context of the original REA model is to look not just at retail enterprises, but enterprises in general, including manufacturing enterprises.

As mentioned above the REA ontology [GM00] extends the original model with transformation events but they are not adequately described, and in examples of REA models they are rarely included and often inconsistently used.

The scope of the REA model has also changed with regards to the levels of enterprise functioning it describes. The entities of the original REA model belong mainly in the *Operational Level* covering day-to-day tasks of i.e. accountants, and the addition of transformations enables modeling of production operations, that are also part of the operational Level. However the Commitments and Agreements of the extended REA model, might take the form of i.e. production plans or budgets and allow the scope to include elements of the *Tactic Level* that involves short-term planning. The *Strategic Level* involving long term planning is beyond the scope of the REA model, and no attempt will be made to include it. Changing the scope from merely registering occurring events after they have happened, to including planned events have an effect on the model and its implementation, as it introduces a change from operating on known data to estimated data.

A consequence of adding transformations and resource planning to the scope is the increased importance of including temporal and spatial dimensions when modeling. When registering resource changing events that have already occurred they are less important, but when planning transformations all the necessary resources have to be available at the exact same time and place.

The first part of this paper aims at providing a better understanding of the semantics of transfers and transformations, including the differences between them and ways to classify and describe them. The second part of this paper presents a property driven approach to creating and implementing REA-based business models. The ideas are at a very preliminary stage and the feasibility of the approach has not yet been determined. However as the intention with this workshop is to discuss new ideas and ongoing work, we have taken the opportunity to include it as an interesting alternative to traditional datamodeling approaches. The REA model will not be described in detail here, as all the topics of the workshop evolve around the REA model, and readers are expected to be familiar with the REA ontology. For readers not familiar with the REA model [GM00] gives a good introduction.

Semantics of Economic Events in the REA Model

This section presents an attempt to provide more details of the ontological definitions of events, their semantics and the constraints that apply to their combinations.

Basic Event Types

The extended REA ontology [GM00]with basic entities and relations as shown in figure 1 defines *transfers* and *transformations* as the two basic event types. Furthermore it distinguishes between *inflow* and *outflow* events, and states three axioms that constrain their *stock-flow* relations to Resources and their *participation* relations to Agents. These definitions are stated as if they relate to both types of events in the same manner, but a closer examination shows that there are some fundamental differences between the two event types, and that it might not be possible to state the axioms in a manner that fits both. This section gives an overview of the characteristics of economic events.

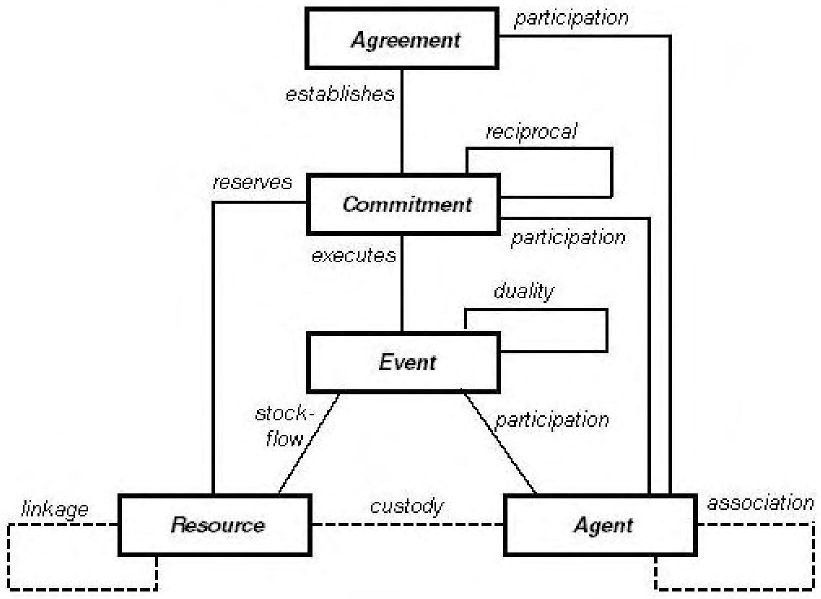


Figure 1: Entities and relations of the basic REA model

What is a transfer ?

According to the ontology [GM00] a transfer creates value through *market transaction with outside parties* and the two stock-flow relations associated are *give* (outflow) and *take* (inflow). In this version of the ontology agents are seen as either *inside* agents or *outside* agents. The common interpretation of these terms are as *internal* or *external* with regards to the enterprise in focus (which is itself an upper level agent composed by other agents i.e. departments and employees). Transfers typically take place between agents from different enterprises and have one internal agent and one external agent participating. But sometimes transfers within the boundaries of an enterprise take place. In this case the ontology's description of the terms *inside* and *outside* are to be interpreted in the sense *providing* or *receiving*. This non-symmetry of using sometimes one definition and sometimes the other leads to confusion when creating and interpreting models, and it causes problems if model evolution leads to changes like i.e. outsourcing parts of an enterprises activities. In this case some outside agents (in the sense ”receiving”) become inside agents (in the sense ”internal”) and vice versa. In this paper we will use the neutral terms ”receiver” and ”provider” as suggested in [Jaq03].

Another issue to consider is what is transferred when a transfer event occurs. Typically *ownership* of a resource will be transferred. If the scope of the REA ontology is within a single enterprise there is an implicit ownership relation to all resources modeled. If a resource is transferred to an external agent it disappears from the view of the model. There is however an explicit *custody* relation that can be interpreted as showing what agent has the rights and physical possession of a resource to transfer or transform it. If the scope of the ontology is enterprises in general a more detailed definition of transfers will be required. In general a transfer may be of one or more **rights** or **responsibilities**. Examples may be *rights of use* and *responsibility of maintenance* for rentals, *liability* as with insurance responsibilities during i.e. transportations. Music and other types of entertainment are usually sold with rights of use but limited rights of copying etc. Unless an agreement states otherwise transfer of ownership means transfer of all associated rights and responsibilities, but it is also possible to trade only some rights to a resource in an exchange.

What is a transformation ?

The REA ontology states that transformations are ”exchanges” that *create value through changes in form or substance* and consist of events that *produce* something (inflow) paired with dual events that *use* and/or *consume* something (outflow). When defining the terms use as *”using up resources in a way that cause them to disappear or leaves their form unrecognizable”* and consume as *”in making use of a resource in a way that leaves the original form discernible”* Black & Black from 1929 [BB29] p.30 is cited. The text referenced does not specify these exact terms and the common way to interpret them is the other way around. When using the Merriam-Webster Online Dictionary (www.m-w.com) consume is listed as synonymous with ”to do away with completely” and use is synonymous with ”the act or practice of employing something”. In this paper the terms are therefore used in the sense that consuming a resource is form-changing or destroying, while using a resource preserves form and identity.

Changing ”form or substance” is not the only way in which the value of economic resources can be changed by transformation. In [BB29] creating value through transformations is described as increasing the ”want-satisfying power” of resources and thereby their value in a sales situation. Three ways of altering the ”want-satisfying power” are mentioned in [BB29] p. 27. One is indeed changes in **form or substance** as the REA ontology mentions. But also changes in **placement or location** as well as changes in **time** are mentioned. As an example of how changes in the placement affect the value of resources can be cited *”A commodity which is in Madagascar when all the people who want it are in Europe has no want-satisfying power unless it can be transported to Europe”*. Transportation is thereby also a type of transformation that changes the spatial placement and value of a resource - without changing its form or substance. The effect of time on the value of a resource is illustrated by the fact that the value of a crop being harvested decreases rapidly, if it cannot be sold before it rots. Likewise the value of good wine can increase with time and Christmas decorations have significantly higher want-satisfying power in December than in June. Age and location as mentioned here are two examples of properties of a resource that can be changed to increase or decrease the value of the resource. But many other properties can be changed to affect the value of an item. I.e. a broken bicycle can be fixed, a recycled bottle can be cleaned, and an employee can be educated to give his or her services a higher value and so forth. So stock-flow relations called move, store, educate, clean and so on would all make just as much sense as use and consume. They would all indicate a change in some property of a resource as well as the value of the resource. Most of these changes will be changes of an identifiable physical property, but i.e. education can be seen as a non-physical value increasing transformation.

The only transformation event causing inflow is the *produce* event that identifies the result(s) of a transformation. The basic subtypes of transformations causing outflow could be stated as:

- **Use** of an economic resource, preserving form and identity. Black & Black call these types of resources ”fixed”.
- **Consumption** of an economic resource destroys it completely or changes its form or structure in a way that leaves it unrecognizable. Black & Black call these types of resources ”circulating”.
- **Transportation** modifies the location of a resource.
- **Storage** of a resource does not necessarily change the form of the resource in any way, but its value is changed due to the time. Christmas decorations are an example. Some resources may also have properties that can be changed by age.
- **Modification** of some specific property of a resource. I.e. *Repairing* a broken bicycle, *Painting* a car or *Educating* personnel in order to increase the value of the work they provide. These types of transformations change a single identifiable property of the resource.

Dualities of events

A key element of the REA model is the *duality* relation that shows which events form an exchange. It is not clearly stated in the ontology, but we make the assumption that only events of the same type (either transfers or transformations) can be connected by dualities. Following the discussion of the differences between the event types above, this seems reasonable as it would make no sense to exchange an altered relation (transfer) with a change of a physical property (transformation). The ontology used the term *Exchange* for dual events of both types. We prefer to introduce the term *Conversion* to distinguish the two from each other. This leaves Exchange as the term for dual transfer events and Conversion as the term for dual transformation events.

Another reason for introducing the Conversion is that there is a fundamental difference between the semantics of dualities connecting transfers and transformations. The inflow (produce) and outflow (use, consume, etc.) events in a Conversion are connected by dualities showing that these ”events” are *all parts of the same overall event or Conversion* as shown by the shaded area in the bread-baking example of figure 2. None of these events make sense to observe in isolation, they represent ”parts” of events taking place simultaneously and one cannot take place without the others. Some of the partial ”events” cause a decrement of resources and some cause an increment.

On the other hand dual transfer events may very well happen at different times and places and can be observed in isolation. In this case the incompleteness comes from the fact that a single event, i.e. a cash receipt (an increment of resources) represents only one side of a transfer. If a global perspective of the scenario modeled is assumed, the enterprise making the payment is included, and the other half of the transfer would be seen as cash disbursement event decrementing resources within that enterprise. The event recorded in one enterprise as a disbursement and in the other as a cash receipt is in fact the same event (shown by the shaded areas of figure 3, and in a global view they may be collapsed into a single ”transfer cash” event. If a duality across the boundaries of enterprises was shown the duality would

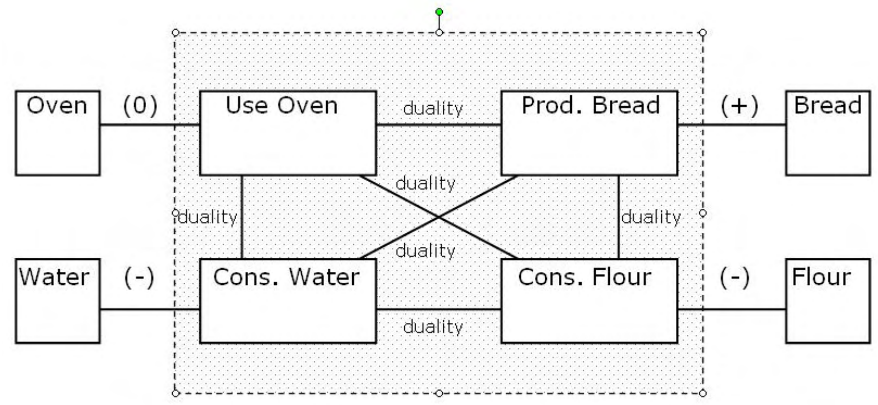


Figure 2: Dualities combining transformation events in a Conversion

be similar to the one used between transformations, connecting increments and decrements caused by the same event. However this is not how dualities are used between transfer events. They are used to show what the cash receipt was a payment of, i.e. the ”Sell Item” event.

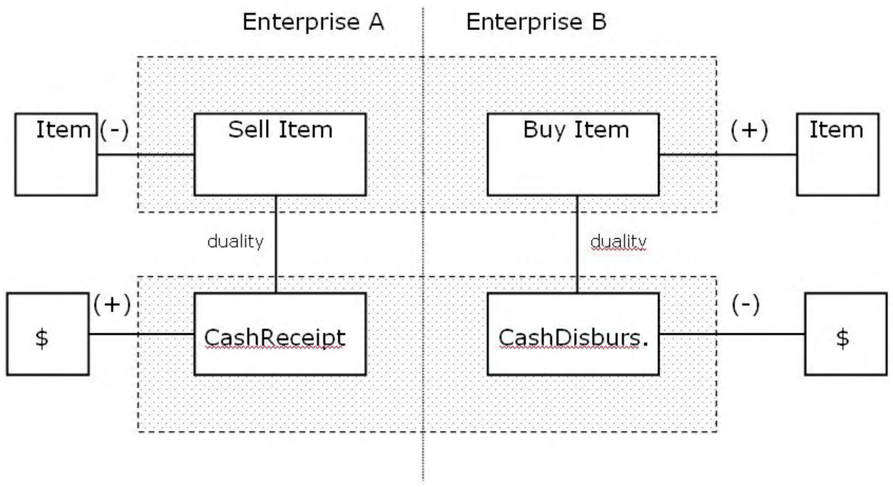


Figure 3: Dualities combining transfer events in an Exchange

The notion of *congruent* events is introduced in [GM00], describing multiple dual events that occur at the exact same space in place and time. An example of congruent transfer events might be a sale in a shop where both agents are present, and hand over the payment and the purchased goods at the same occasion. Transformations are not mentioned explicitly, but as argued above, they must be congruent by nature.

Axioms

The REA ontology [GM00] states three axioms to be observed for all model instances. They are defined as follows:

- **Axiom 1**: At least one inflow event and one outflow event exist for each economic resource; conversely inflow and outflow events must affect identifiable resources.
- **Axiom 2**: All events affecting an outflow must be eventually paired in duality relationships with events affecting an inflow and vice-versa.
- **Axiom 3**: Each exchange needs an instance of both the inside and outside subsets.

The inside and outside subsets mentioned in axiom 3 can be interpreted as internal vs. external agents with respect to the enterprise in focus.

The first axiom makes sense for both transfers and transformations. The second axiom is true for all transformations as resources cannot appear or change state out of the blue. Often a transformation involves at least three dual transformation events. I.e. production of goods by consumption of raw materials, and use of labor and machinery. For transfers the second axiom is often true, but as described by McCarthy in [2] for practical reasons single events with no dualities might be the best way to describe gains and losses and matched expenses like i.e. tax, heating and maintenance. The third axiom is not quite clear because of the different possible meanings of inside and outside. If inside/outside is meant in the sense internal/external it clearly only concerns transfers as transformations will only involve internal agents. If it is meant in the sense provider/receiver it might apply to both types of exchanges, with the addition that for transformations only, a single agent may be both the receiver and the provider.

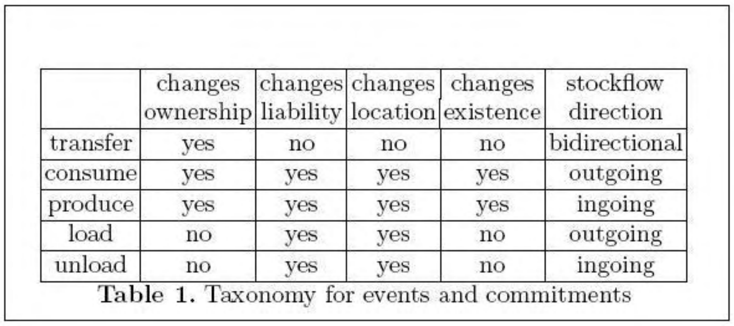
It seems that the three axioms and the general descriptions of Exchanges and Events were made primarily with transfers in mind. The conclusion here is that there are differences between the two Event types that need to be observed, and that the nature of transformations is inadequately described by the ontology. A summarized overview is shown here:

Property	**Transfers**	**Transformations**
Are coupled by dualities in	Exchanges	Conversions
Are always	Across boundaries of agents	Within boundaries of an agent
Agents participating	Two, of competing economic interest	At least one
Dualities connect	Separate events	A congruent event
Temporal nature	Are instant	May have a duration
Change	Agent-Resource relations	Properties of a resource

Structured approaches to describing the REA entities

The analysis of semantic details of the REA ontology presented here, has so far been unstructured and intuitively guided by common reason. But there have been proposed more structured approaches to describing the semantic details of events. One example is in [GM02] where a mapping of the concepts of the REA ontology to Sowas ontological categories [Sow00] is proposed. A different type of approach to establishing structure and consistency of the model, could be to apply a variant of the classical data modeling principles using CRUD diagrams[1]. As the only actions within the scope of the REA model are economic events affecting the *state of resources*, resources are the primary REA entities subject to applying the CRUD principles on. The users management of agents, events etc. are necessary parts of a REA-based implementation, but not part of the value-flows modeled. Furthermore the ”R” (Read) from CRUD can be omitted as retrieval of information from the system is not modeled explicitly. That leaves us with investigating Creation, Deletion and Updates (modification) of resources as the essential events in a REA model. Creation and Deletion affect the existence of resources and correspond to the event types *produce* and *consume*. Modification on the other hand represents changes to the state of a resource and does not affect existence or identity. Whether transfer events affect existence or not, depends on the perspective the model is seen from. If a dependent (trading partner) view is assumed one may choose to model resources as being created/deleted when purchased or sold. If an independent (global) view is assumed the resources sold do not cease to exist, but merely change owners.

In [Hes06] a classification scheme is proposed, describing the effects of event types involved in a model for the SCM/VMI enterprise domain. It shows which of the resource properties as *ownership*, *liability*, *location* and *existence* are affected by the specific event types.



	changes ownership	changes liability	changes location	changes existence	stockflow direction
transfer	yes	no	no	no	bidirectional
consume	yes	yes	yes	yes	outgoing
produce	yes	yes	yes	yes	ingoing
load	no	yes	yes	no	outgoing
unload	no	yes	yes	no	ingoing

Table 1. Taxonomy for events and commitments

Figure 4: Table 1 of [Hes06]

The events listed form a general taxonomy of economic events in the given domain. When implementing specific solutions additional columns and rows may be added. This is one modeling approach where one event can affect several properties. A different approach would be to determine the semantics of events by using simple minimal events that each modify one property, and use these as small building blocks to put together more complex domain specific events like i.e. ”load” that changes both

[1]CRUD is an acronym for Create, Retrieve (or Read), Update and Delete (or Destroy). These are the actions that can be performed on data in a database and diagrams showing data vs. actions (C, R, U or D) can be used to describe what actions users are allowed to perform

location and liability. Note that events that change existence always affect all properties when shown this way.

Describing the same effects on resources and their properties using C(R)UD principles could be done as:

Transfer	U (ownership)
Consume	D
Produce	C
Load	U (liability, location)
Unload	U (liability, location)

This is an example of describing the semantics of effects of selected events from a specific restricted domain to a list of specific resource properties. The same principles of using the C(R)UD approach on relevant properties of resources, can be applied to any specific business case or domain. The following section gives an overview of how this can be done in a uniform manner based on classifications of the relations between the properties relevant to the model, and the ways they can be modified by events.

A Property Driven Approach to Applying the REA Ontology

Traditionally when the REA ontology has been applied to a business case as described in i.e. [DCH05], focus has been on which REA entities to model. What are the resources, agents and events present and necessary to record in that specific enterprise? Lets assume a different approach. As the aim is to model value flows within and to and from an enterprise, resources are a key entity, so lets start with them. Resources are affected by events. But the REA ontology doesn't say much about the semantics of events. What actually happens when an economic event occurs? The answer is that either a resource is added to or removed from the system, or some property of a resource is changed. If an item is sold the ownership is changed. If a car is painted the color and/or level of maintenance is changed, hereby changing its value. [GM99] states that ”*Value is defined as a deliverable portfolio of product or service attributes attractive to the firm's ultimate customers.*”. *So besides from exchanges, changes to the attributes or properties of resources is how the value within an enterprise can be increased.* When we use the term ”properties of a resource” it can be seen as synonymous with the terms ”attributes” or ”characteristics” often used in REA literature. We use the term in a slightly broader sense to call i.e. ownership a property, even if it may be thought of as a relationship rather than an attribute of a resource.

Various Types of Properties

Determining the characteristics or properties of a resource is a somewhat philosophical question, but in the context of REA models an answer to the question of whether or not to include a property, may be determined by considering whether or not changes to that property affect the value or availability of the resource. Specifying properties of resources is also closely related to determining the Resource Types of the Knowledge Level in a REA model using typifications and groupings as described in i.e.

[GM03]. The subject of typifications will not be addressed in detail in this paper, but the separation between static vs. dynamic and intrinsic vs. extrinsic properties described in the following seems essential in that discussion as well.

The REA ontology employs the differentiation between the *Operational Level* and the *Knowledge Level* as suggested by Martin Fowler in [Fow97]. At the *Operational Level* indentifiable instances of events, agent and resources are described and at *Knowledge Level* the notion of *Type Images* is used to describe the correlations between entities that exist, regardless of whether or not specific instances of the entities exist. In [GM00] p. 13 the example of a lion being a roaring member of the cat family is mentioned as an example of a relation that applies to all lions and continues to exist even if the lion race was to become extinct.

Based on this description it would seem reasonable to conceive the notion of typification in REA as a persistent segregation into static type hierarchies as known from i.e. the concept of inheritance in object oriented modelling. In this case a lion would be one subtype of a roaring cat animal, which again is a subtype of a cat animal, which is a subtype of a mammal and so forth, with each subtype inheriting the properties of its supertypes.

On the other hand additional examples from the REA domain are mentioned that don't fit this interpretation. One is the example of agent rankings where customer types (small, medium, big) and the experience of sales personnel (experienced, non-experienced) can be determined and matched. These typifications are not static, but can change dynamically during the lifetime of the agent entities. This seems to be rather like a way of specifying metadata that describe certain properties of the agents. One might say that ”small” ”medium” and ”big” are the possible values of a ”customertype” property of a Customer. Note also that these properties are exclusive - exactly one must be selected. Other similar properties as i.e ”locationtype” being ”domestic” or ”international” can be specified. This is also an indication that implementation by inheritance is not a good solution as this would lead to multiple inheritance.

Another example of typification from [GM00] is the distinction between the event types ”Specialty Store Sales”, ”Mail Order Sales”, and ”Internet Sales”. In this case the typification is indeed static due to the occurrent nature of events as opposed to agents that are continuant. This means that events have no dynamic properties and that inheritance could be used (although not necessarily recommended) for describing economic event types, as opposed to resource types and agent types.

Property Driven Modeling

The idea of using properties as a starting point for developing REA models is based on the central role of resources. If the essential issue of using REA to model enterprise systems, is to record the fundamental data of economic events, why not focus on what changes are made to the state of resources (the desired result of events) instead of i.e what processes currently take place in the enterprise? That way the structure of data recorded is not biased by the way it is currently used, and may possibly be useful in other areas of the enterprise functions.

The first steps of applying a property driven approach could be:

1. Identify relevant resources.
2. Identify their value-affecting properties.
3. Identify Buy/Produce and Sell/Consume events for all resources (fulfilling Axiom 1).
4. Identify relevant Modify events for all value-affecting dynamic properties.

Thus basing the selection (and naming) of events on the properties they modify, not the processes they take part in. Thereby the flexibility, adaptability and possibility of reusing data may be increased. Using the C(R)UD notation shown above to classify the events modeled and their effects gives a good overview of whether or not any events are missing.

The principles of identifying changes to properties may be a good tool when modeling solutions including transformations and conversions. In the case of a retail enterprise where transfers are dominant there would be not benefit of using this approach as steps 1 and 4 mentioned above are irrelevant.

Detaching Properties

When dealing with ontological engineering some challenges and decisions are often left open for interpretation as ”implementation issues”. It may however be beneficial to use the feedback obtained from implementing specific solutions to improve the ontologies as argued in i.e. [BS04]. How the attributes or properties of a resource are modeled and implemented have an impact on the flexibility and evolvability of solutions. During the lifetime of an enterprise system there may be changes to the way business is conducted, and how data is organized and connected determines how easy it is to adapt to changes. The ideas presented here are in many ways similar to those of adaptive programming and aspect-oriented programming.

When identifying properties, some are intrinsic to a resource and cannot be observed in isolation from the resource they describe, i.e. the serial number of a car. Others, like the tires a car is currently equipped with, are extrinsic. An interesting approach to take could be to detach properties from the resources they describe to the extent possible. This way the state of properties determining the value of a resource becomes a derivable value determined by the changes that have been made to that property. How to determine the value of the derived property depends on business logic associated with the events affecting the property. This way of registering state changes instead of maintaining a ”current state” value, is very similar to the central idea of the REA model being event-driven and recording i.e. changes affecting accounts and deriving the balances instead of maintaining a ”current balance” value.

If a company producing cars is taken as example, there could be a ”Mount Tires” event adding or changing the set of tires (a property) associated with a car. There might also be a ”Paint” event that adds/changes the color property, and a ”Anti-corrosion treatment” might update the date stating when it was last applied. If properties are detached as individual components related to resources, they and their related events can be reused for several different resources. I.e. a ”Paint” event that modifies a ”color” property and has information on quality, colorcode, and amount of paint to use, may be just as applicable to bicycles or dustbins. It also makes it possible to add new properties and their related events without changing existing resources. I.e. a car company may want to add a ”Has GPS” property to the features of a car. This is done easily by adding a ”Has GPS” property and a ”Mount

GPS” event, and allowing them to apply to relevant car types. Likewise if the car company decides to start a rental business an ”Available for rent” property that depends on the most current of the two event types ”Rented” and ”Returned” may be added.

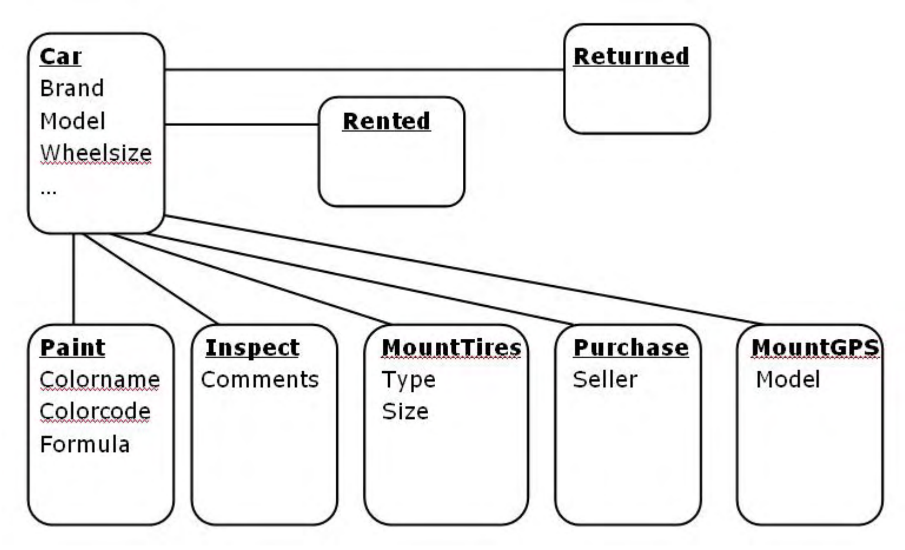


Figure 5: Different types of events affecting properties that may be associated with a car

Thus events may affect one or more properties, properties may depend on one or more events, and property-changing events may either accumulate or cancel previous changes (i.e. painting).

Conclusions

We have presented a non-formal but detailed description of the semantics of economic events in the REA model. In particular the difference between transfers and transformations and the dualities that connect them have been described. We have argued that the existing axioms of the REA ontology need to be modified and possibly extended in order to take the differences of the event types into account. A structured way of describing the effects of events on resource properties has been demonstrated using C(R)UD principles, and a property oriented way of approaching modeling conversions and implementing properties of resources has been outlined. The ideas presented are at a very preliminary stage and the possible benefits and drawbacks of using a property oriented approach are yet to be determined. But as shown the benefits may be seen in both the levels of modeling and implementation and the approach could possibly lead to increased flexibility, adaptability and consistency of solutions.

References

[BB29] John D. Black and Albert G. Black. *Production Organization*. Henry Holt and Company, xi edition, 1929.

[BS04] Signe Ellegaard Borch and Christian Stefansen. Evaluating the REA Enterprise Ontology from an Operational Perspective. *Proceedings of the EMOI-INTEROP'04 - Enterprise Modelling and Ontologies for Interoperability Workshop. Published in proceedings of the CAiSE 2004 Workshops*, 3, 2004.

[DCH05] Cheryll Dunn, J. Owen Cherrington, and Anita S. Hollander. *Enterprise Information Systems: A Pattern-Based Approach*. McGraw-Hill, 3rd edition, 2005.

[Fow97] Martin Fowler. *Analysis Patterns: Reusable Object Models*. Object Technology Series. Addison-Wesley, Reading, Massachusetts, 1997.

[GM99] Guido Geerts and William E. McCarthy. An Accounting Object Infrastructure For Knowledge-Based Enterprise Models. *IEEE Intelligent Systems and Their Applications*, pages 89–94, July-August 1999. http://www.msu.edu/user/mccarth4/x4xop.lo.pdf.

[GM00] Guido L. Geerts and William E. McCarthy. The Ontological Foundation of REA Enterprise Information Systems. http://www.msu.edu/user/mccarth4/Alabama.doc, August 2000.

[GM02] Guido L. Geerts and William E. McCarthy. An ontological analysis of the economic primitives of the extended-REA enterprise information architecture. *International Journal of Accounting Information Systems*, 3:1–16, 2002.

[GM03] Guido L. Geerts and William E. McCarthy. Type-Level Specifications in REA Enterprise Information Systems. Working Paper, Michigan State University, http://www.msu.edu/user/mccarth4/UTS-seminar/Type%20paper%20final%20submission.doc, 2003.

[Hes06] Anders Hessellund. Supply Chain Modeling with REA. TR 2006-80, The IT University of Copenhagen, Denmark, 2006.

[Jaq03] Mette Jaquet. REAlistic: En REA-model uden perspektiver. Masters Thesis written at the IT-University of Copenhagen, Denmark, March 2003.

[McC79] William E. McCarthy. An Entity-Relationship View of Accounting Model. *The Accounting Review*, LIV(4):667–686, October 1979.

[McC82] William E. McCarthy. The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment. *The Accounting Review*, LVII(3):554–578, July 1982.

[Sow00] John Sowa, editor. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks Cole Publishing Co, 2000.

Towards an Operational REA Ontology Using Web Ontology Languages

Frederik Gailly[♣], Geert Poels

Management Informatics Research Unit

Faculty of Economics and Business Administration

Ghent University

Frederik.Gailly@Ugent.be, Geert.Poels@Ugent.be

POSITION PAPER

Research Goal, Background and Motivation

Ontology engineering is a promising direction for information systems research and practice. The use of ontologies can improve communication between people and organizations, can create interoperability between systems, and can improve the reusability and reliability of the systems engineering process[1]. Many researchers have recognized these opportunities, but the adoption of ontology applications in practice is still low. This lack of use in practice can be partly attributed to the scarcity of well-formalized ontologies. The last five years the World Wide Web Consortium (W3C) has published a series of recommendations about XML technologies that offer support for using ontology in practice. Nowadays stable ontology languages like RDF and OWL can be used for the formalization of ontologies. More recently languages were proposed that make it also possible to query ontologies (SPARQL and OWL-QL), which is a requirement for realizing Semantic Web applications.

Despite the availability of supporting languages and technologies, business domain ontologies are generally underspecified and insufficiently formalized for practical application purposes. Analogous to traditional software development, quality problems with proposed ontologies are often the result of a poor development process. Our research project aims to improve the development process of business domain ontologies, which should result in ontologies that can be applied in practice for a variety of business applications. Our target business domain ontology is the Resource Event Agent ontology. The REA ontology is based on the REA accounting model developed by McCarthy [2] and was further extended into a comprehensive enterprise information architecture by

[♣] Corresponding author

Geerts and McCarthy [3]. Other business domain ontologies that could be used as case-studies in our research are the Toronto Virtual Enterprise Ontology (TOVE) [4], the ENTERPRISE ontology [5], and the Business Model Ontology [6].

Geerts [7] has explored how XML technologies can be used for the operationalization of the REA ontology. According to Geerts a business domain ontology becomes operational when its specification is recorded and actively used by applications. Geerts [7] defines an enterprise system architecture based on XML technologies for supporting an operational REA ontology. The components of the conceptual enterprise system architecture are the enterprise data, the enterprise schema that contains enterprise-specific information structures and business rules, and the enterprise ontology which imposes the 'enterprise domain' –specific REA structure on the enterprise schema. The implementation part of the architecture separates these three components by codifying them in different XML or XML Schema documents. Separately describing the three components allows XML validation and transformation technologies to be used to verify compliance of the enterprise data against the enterprise schema and the enterprise schema against the REA ontology. We consider the XML-based REA enterprise system architecture of Geerts (2004) a useful framework and starting point for our research. We extend and adapt this framework in two directions.

First, the enterprise system architecture proposed by Geerts corresponds to a large extent to the single ontology approach that can be used for ontology-based data integration [8]. The single ontology approach (see figure 1-i) means that different source schemas are directly related to a global ontology that provides a vocabulary of business concepts and an axiomatic structure of relationships between these concepts. The major drawback of this approach is that all source schemas have nearly the same view on a domain, with the same level of granularity. In order to realize this approach a certain degree of standardization is needed and its our opinion that this will be very hard to realize in business practice. Another possible approach for realizing ontology-based data integration is the multiple ontology approach (see figure 1-ii), where each data source is described by its own local ontology and instead of using a common ontology, local ontologies are mapped to each other. The third and probably most feasible approach is the combination of the two preceding approaches. The hybrid ontology approach (see figure 1-iii) builds a local ontology for each data source schema which is mapped to a global shared ontology. The REA ontology could now be a local ontology (or one of many local ontologies) for (part of) an enterprise, and data integration can be achieved by reference to the shared ontology.

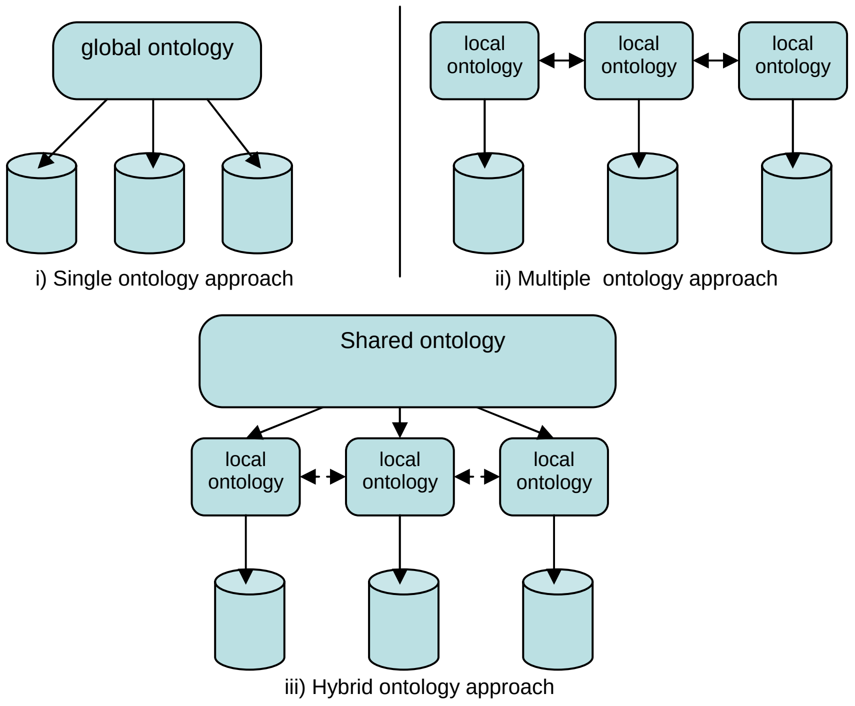


Figure 1: Ontology-based Information Integration Approaches

Second, we argue that the implementation part of the REA enterprise system architecture should include ontology languages for the formalization of the ontologies. It is true that the difference between XML Schema and web ontology languages is small, but we follow the view of Klein et al. [9] which see XML schemata as a means to provide integrity constraints for information sources and ontology languages as a means to specify domain theories. Domain theories consist of domain rules which must be able to express integrity of data but also the domain conceptualization and therefore must have a greater expressive power [10]. Using ontology languages also means that the ontology can be more easily mapped onto another ontology which is for instance necessary when the ontology is used in a hybrid ontology approach for data engineering (e.g. mapping a local ontology to the global, shared ontology). As a result we adapted Geerts' architecture and stipulated that for the implementation (formalization) of the ontological primitives, enterprise concepts and ontological logic of the REA ontology we intend to use ontology languages like RDFS and OWL. For the implementation of the enterprise schema XML Schema, RDF or even the instantiation constructs of OWL could be used. Important at this level is that the proposed schema corresponds to the formalized ontology. Figure 2 represents an overview of our adjusted enterprise system architecture based on the work of Geerts.

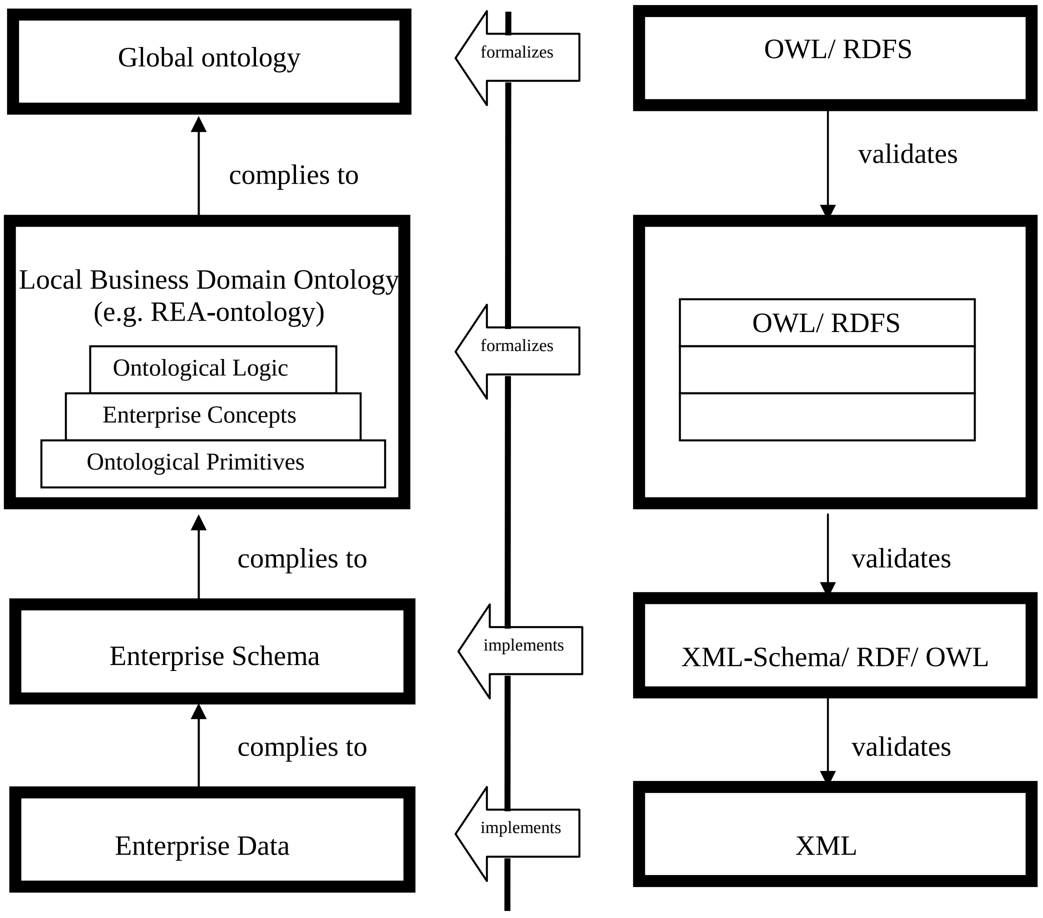


Figure 2: Operationalisation of the REA-ontology (based on [7] p81)

APPLICATIONS

Today the REA ontology is mainly used as a guide during the software engineering process and in this sense it can be used as a quality assurance tool when developing enterprise systems. The REA ontology provides the basis for various business process reference models and thus assists the identification of requirements and helps assuring the correctness, relevance and completeness of the requirements specifications.

One of the challenges of today is to use business domain ontologies to give semantic meaning to enterprise data and as a result make it easier to create interoperability within and between enterprise applications. Web ontology languages make it possible to implement the REA ontology as a local ontology for a business or business application and as a result give semantic meaning to enterprise data. Additionally the local REA ontology can be mapped to a global ontology which provides a common framework that allows data to be shared and reused across application and enterprise boundaries.

The use of the REA ontology in real applications is still very limited. Borch et al. [11] explored how their REA ontology formalization in XML could be used as a platform independent model in the Model Driven Architecture (MDA) approach to the development of accounting systems. Other proposed applications of the REA ontology are only theorized but are to our knowledge not illustrated with real applications. Haughen and McCarthy [12] state that the REA ontology is perfect for multi-company supply chain collaboration and REA can offer a standardized semantic model that can actually support all supply chain activities. During the first International REA Technology Workshop different authors have also proposed possible applications for the REA ontology: automated reasoning about business processes, and valuation and optimization of processes , automating processes across supply-chains, but they all agree that a formal representation is necessary [13].

DESIGN APPROACH AND CURRENT STATE OF THE PROJECT

One of the first problems that must be solved to achieve a successful operationalization of the REA ontology is the development of a good formal representation of the ontology. A series of methodologies for ontology engineering have been proposed by the Artificial Intelligence community. One of the most complete ontology engineering methodologies is METHONTOLOGY, which in accordance to the IEEE standard for software engineering distinguishes four different activities in the ontology development process: specification, conceptualization, formalization and application [14]. The METHONTOLOGY development process has also been extended for the reengineering of ontologies [15] and identifies three main activities: reverse engineering, restructuring and forward engineering (see figure 3).

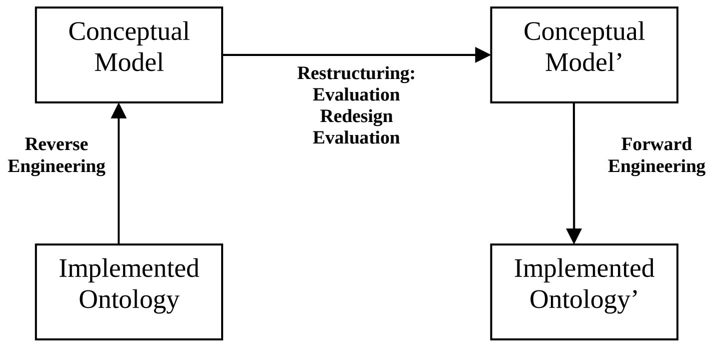


Figure 3: Ontology reengineering process

This reengineering methodology can use techniques described in the METHONTOLOGY development process (see [16] for an overview), but specific for the reengineering of business domain ontologies we think it is important to include a top-level ontological analysis. It is our believe that a further integration of existing top-level ontologies like SUMO [17], BWW [18] and

SOWA [19] into the development process could further improve the business domain ontologies and will make it easier to create interoperability between different systems because it makes it easier to compare and map the different local ontologies or define a global ontology.
Furthermore it is also important to use existing data modeling languages during the conceptualization of a business domain ontology. Data modeling languages like ER and UML are generally known by business people and this will make it easier to communicate and discuss the business domain ontology. However some issues need to be taken into account when using data models as conceptualizations for ontologies. Unlike data models which are developed for a specific application, ontologies need to be as generic and task-independent as possible [10]. Ontology engineers must also follow an Open World Reasoning approach during the development of the domain ontology. As opposed to data modeling, in ontology engineering something is false only if it can be proved to contradict other information in the ontology. In most cases data modeling languages like ER and UML will not be able to express all domain rules and constraints. For UML this can be partly solved by the incorporation of OCL but for more sophisticated ontologies even the expressive power of OCL will be too limited. Including a top-level analysis and using data modeling languages for the conceptualization will also facilitate the forward engineering process because it makes it easier to select the appropriate formalization rules and could also in the long run automate the formalization process. For instance, UML-to-OWL mapping rules could be used. For the moment no uniform approach exists but the Ontology Definition Metamodel (ODM) proposal of the Object Management Group (OMG) intends to offer such an approach [20].
In the first faze of our research project we adopted this reengineering methodology for the development of a formal representation of the basic transaction pattern of the REA-ontology. The REA ontology does not contain a widespread accepted formalization and as a result the original conceptualization was used as a starting point. Geerts and McCarthy[3] have used mainly text for describing the REA-ontology and have added some graphical representations in some traditional data modeling languages like ER and UML class diagrams.
The restructuring of the existent conceptualization resulted in natural language definitions of the basic concepts and their relations. An ontological analysis of the concepts based on Sowa’s ontology was already performed by Geerts and McCarthy [3] and was used as a starting point. In the future we will extend this analysis with other more commonly used high-level ontologies.
Based on the obtained definitions and the original graphical representations, a new reengineered conceptual model of the basic REA pattern in UML was developed (see figure 4). A distinctive characteristic of the REA-ontology which is not represented in the original conceptualization is the specialization of the relations between the different concepts. For instance the stock-flow relation is specialized in inflow and outflow relations which can be further specialized in use, consumption, give (for outflow) and take and production (for inflow). It is our opinion that the relations should be

represented with association classes which can be further specialized. Specializing the different relations opens up the possibility of using OCL for the representation of ontological axioms, e.g. the basic REA axiom stipulating that all (types of) events effecting an outflow of economic resources must be paired in duality relationships with (types of) events effecting an inflow and vice-versa [3].

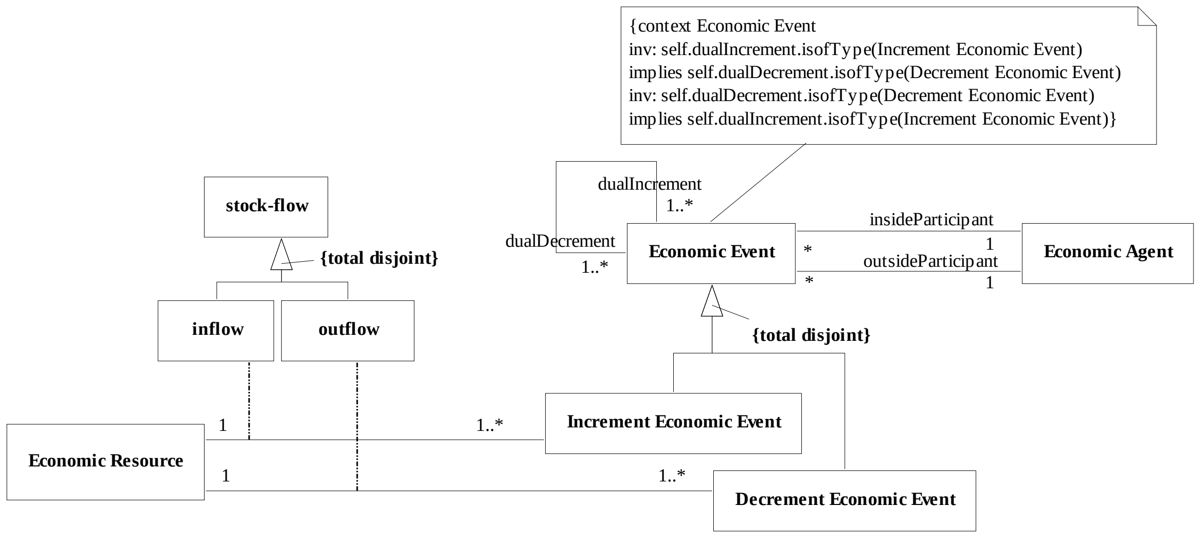


Figure 4: Reengineered conceptual model basic REA pattern

The tentative ODM proposal is used for the formalization of the basic REA pattern. Some of the transformations from UML to OWL are very straightforward, but even in this small part of the REA-ontology it is sometimes hard to determine the most appropriate mapping. According to the guidelines, UML associations can be mapped into OWL object properties, but during the formalization of the REA UML class diagram we came across some problems. The mapping rule is stipulated for associations that are directed and where the associations are labeled by properties. This is not the case in the REA UML class diagram where we have two sorts of associations: association classes and labeled bidirectional associations. We hope that future ODM developments will solve this issue. A full overview of the formalization of the basic REA pattern will not be presented in this paper, but can be consulted at http://users.ugent.be/~fgailly/phd.

NEXT STEPS

The final objective of this research project is the operationalization of existing business domain ontologies. The first thing that needs to be done is improving the quality of the existing ontologies. This can be partly achieved by using recent ontology engineering methods and tools and should eventually result in better formalized ontologies which can be used in practice.

At this stage we have developed an ontology reengineering methodology specific for business domain ontologies which we plan to extend in a number of ways. Primarily the proposed methodology needs to be further elaborated and described. More specifically the ontological analysis of the concepts needs to be further investigated for some existent high-level ontologies and the use of data modeling languages for the conceptualization of ontologies should be further explored. Afterwards the proposed methodology will be applied for the reengineering of the complete REA-ontology and for other existing business domain ontologies.

At a later stage the obtained and improved conceptualizations and formalizations are going to be compared and evaluated. During this analysis emphasis will be evaluating the feasibility of already proposed applications and identifying possible new applications of the domain ontologies. Eventually this should lead to the development of some small scale applications which must illustrate the usefulness of business domain ontologies for practical businesses applications.

References

1. Ushold, M., Gruninger, M.: Ontologies: Principles, Methods and Applications. The Knowledge Engineering Review **11** (1996) 93-136
2. McCarthy, W.E.: The REA Accounting Model: A Generalized Framework for Accounting Systems in A Shared Data Environment. The Accounting Review **july** (1982) 554-578
3. Geerts, G.L., McCarthy, W.E.: An Ontological Analysis of the Economic Primitives of the Extended-REA Enterprise Information Architecture. International Journal of Accounting Information Systems **3** (2002) 1-16
4. Fox, M.S.: The TOVE Project: A Common-sense Model of The Enterprise. In: Belli, F., Radermacher, F. (eds.): Industrial and Engineering Applications of Artificial Intelligence and Expert Systems. Springer-Verslag, Berlin, Germany (1992) 25-24
5. Ushold, M., King, M., Moralee, S., Zorgios, Y.: The Enterprise Ontology. The Knowledge Engineering Review: Special Issue on Putting Ontologies to Use **13** (1998) 31-89
6. Osterwalder, A.: The Business Model Ontology - a proposition in a design science approach. Ecole des Hautes Etudes Commerciales. University of Lausanne, Lausanne (2004)
7. Geerts, G.L.: An XML Architecture for Operational Enterprise Ontologies. Journal of Emerging Technologies in Accounting **1** (2004) 73-90
8. Wache, H., Vögele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., Hübner, S.: Ontology-based Integration of Information - A survey of existing Approaches. . IJCAI-01 Workshop on Ontologies and Information Sharing (2001)
9. Klein, M., Broekstra, J., Fensel, D., van Harmelen, F., Horrocks, I.: Ontologies and schema languages on the web. In: Fensel, D., Hendler, J., Lieberman, H., Wahlster, W. (eds.): Spinning the Semantic Web. The MIT Press (2003)

10. Spyns, P., Meersman, R., Jarrar, M.: Data modeling versus Ontology engineering. SIGMOD record: Special Issue on Semantic Web and Data Management **31** (2002) 12-17
11. Borch, S.E., Jespersen, J.W., Linvald, J., Osterbye, K.: A Model Driven Architecture for REA based systems. Proceedings of the Workshop on Model Driven Architecture: Foundations and Applications. University of Twente, Enshede, The Netherlands (2003)
12. Haugen, R., McCarthy, W.E.: REA: A Semantic Model for Internet Supply Chain Collaboration. Proceedings of the Business Objects and Component Design and Implementation Workshop VI: Enterprise Application Integration. (2000)
13. Stefansen, C.: Transforming the Resource Events Agents Model into a Formal Process-Oriented Enterprise Framework. First International REA Technology Workshop, Copenhagen, Denmark (2004)
14. Fernández-López, M., Gómez-Pérez, A., Juristo, N.: METHONTOLOGY: From ontological art towards ontological engineering. Working Notes of the AAAI Spring Symposium on Ontological Engineering. AAAI Press, Stanford (1997)
15. Gómez-Pérez, A., Rojas, M.D.: Ontological Reengineering and Reuse. In: Fensel, D., Studer, R. (eds.): 11th European Workshop on Knowledge Acquisition, Modeling and Management. Springer-Verslag, Dagstuhl Castle, germany (1999) 139-156
16. Gómez-Pérez, A., Fernández-López, M., Corcho, O.: Ontological Engineering. Springer-Verslag (2004)
17. Pease, A., Niles, I., Li, J.: The Suggested Upper Merged Ontology: A large Ontology for the Semantic Web and its Applications. AAAI-2002 Workshop on Ontologies and the Semantic Web, Edmonton, Canada (2002)
18. Wand, Y., Weber, R.A.G.: An ontological analysis of some fundamental information systems concepts. In: DeGross, J.I., Olsen, M.H. (eds.): Proceedings of the Ninth International Conference on Information Systems, Minneapolis - USA (1988) 213-255
19. Sowa, J.: Knowledge Representation: Logical, Philosophical, and Computational Foundations. Pacific Grove, Brooks/Cole (1999)
20. OMG: Ontology Definition Metamodel. Object management Group (2005)

REA as an e-business ontology.

Position paper

Fred van Blommestein
Faculty of Management and Organization, Rijksuniversiteit Groningen,
Postbus 800, 9700 AV Groningen, the Netherlands
fred@fl owcanto.com

Summary

REA has been recognized as a sound foundation for an e-business ontology. However, some adaptations are needed to cater for the complex and dynamic context of e-business.

Introduction

The REA ontology was originally designed for business accounting [1]. When e-business standards and implementations developed beyond the mere exchange of electronic messages, it was recognized that REA could very well serve as the basis for inter organizational (e-)business processes as well [2]. The reason is that REA is based on the fundamental principle of the exchange of economic resources, which is the essence of commercial activity.

However, the present REA ontology lacks some concepts and structures that are essential to the (automation of) complex commercial and operational business relations today. In particular:

- REA does not include concepts or mechanisms to define the (legal, regulatory and technical) *context* of a business relation, which defines the boundaries (or the 'playfield' of the bilateral contracts to be negotiated and closed.
- A contract in REA is (by definition) a bundle of commitments to make economic events happen, while in practice many contracts just set the conditions for 'lower level' agreements (like call-offs) that really trigger the events.
- Duality and reciprocity are modeled in REA as one-to-one direct associations between Commitments, resp. Events, while in practice these associations are often many-to-many and are defined and controlled by the contract.
- REA supports the definition of *what* resources are exchanged, but not *when* they are and in what order (the business process). Although process definition is probably out of the REA scope, for e-business systems it is essential and the link with the REA concepts must be clear.
- Commitments in REA are defined in an absolute way, while in practice (and certainly during negotiations) commitments are often conditional, to other commitments to be accepted by the other party, or to events to happen.

In this paper we give a number of suggestions, how to adapt the REA model to support complex e-business relationships.

Business conversation

A business conversation consists of a series of statements or utterances, alternately (but not strictly alternately) uttered by the two parties. Utterances may relate to real world events that a party has observed (like the production of goods or the delivery of consignments) or to decisions the party has made (like the decision to purchase products from the other party). Between the parties there must be some communication system installed that relays the utterances. That system can be as simple as the air (allowing the parties to talk to each other) or be as complex as a network connection between the party's ERP systems. In both cases the conversation between the parties has the same meaning, at least commercially and legally.

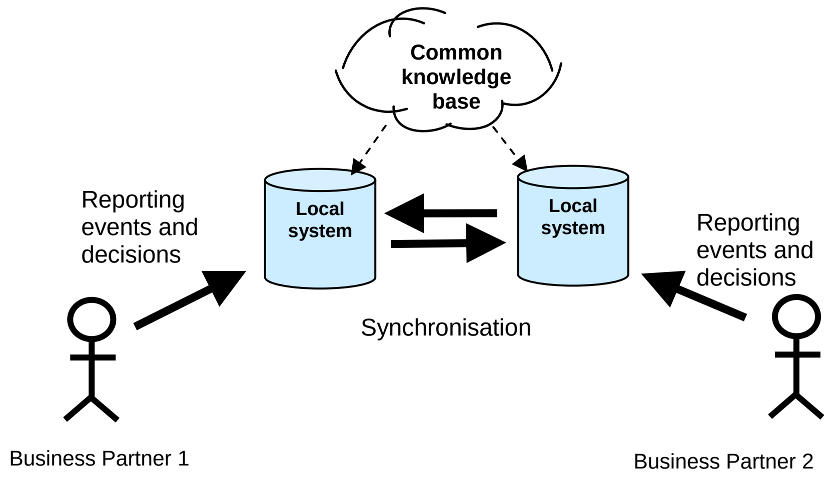


Figure 1. E-business communication

The history of the conversation is significant for the commercial and legal relationship and consequently for the future course of the conversation. Therefore utterances are remembered by the parties or, in other words, (virtually) stored in a knowledge base that is common to the parties. Technically that "knowledge base" may be implemented in the respective private information systems or ERP-systems of the parties. The knowledge base may alternatively (or additionally) reside in the file cabinets in the parties' offices, where paper documents are being stored. In case the conversation is conducted by telephone or during a business lunch the knowledge bas may even only be the personal memories of employees or agents.

Utterances make reference to the knowledge base and their meaning is dependent on the state of affairs in that base. Therefore there must be consensus among the parties about the ontological structure of the knowledge base. The REA model forms a sound basis for that structure. On the other hand, the knowledge base in fact only contains utterances that were uttered earlier during the conversation. In principle the ontology itself must be defined by means of utterances and accepted by the other party.

Not all utterances are allowed at any time in order to be (economically, commercially, legally or even socially) meaningful. The sequence of utterances (and their content) must fit a pattern that is agreed among the parties in advance. Such agreement may have been reached in explicit bilateral negotiations, or it may implicitly be defined in the context of the industry the parties do business in. Patterns and conditions may also be imposed by legislation. In fact a hierarchy exists, where (inter)national legislation defines the playfield where industry sectors fill in general patterns and conditions. Individual organizations bilaterally specialize those patterns and conditions in contracts. See figure 2.

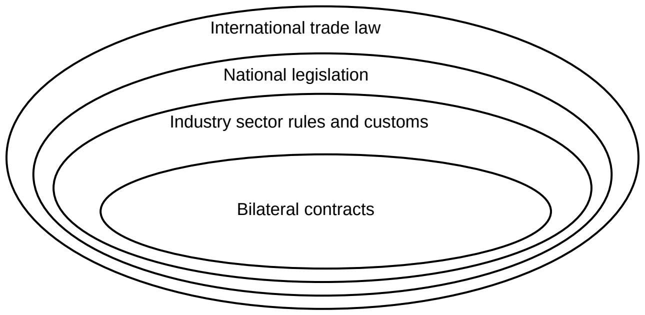


Figure 2. Hierarchy of communication patterns and conditions.

Utterances have some purpose in the business relationship; they control the economic events parties need to perform in order to trade successfully. Decisions and events result in updates to the knowledge base. As knowledge is added, subsequent events and decisions are triggered, defined or restricted. Future conversation patterns may be restricted as well. In this way processes can be specialized. The ‘playground’ of the business relationship is restricted as illustrated in figure 2, by means of a normal business process (e.g. contract negotiation and conclusion).

Linguistically, an utterance has a specific structure. An utterance assigns values or value spaces to properties of (real life or virtual) objects or object classes. The assignment however is performed subjectively by a party, with some intention. So each value(-space) assignment has some intentional value that may be expressed by a verb. Jayaweera [] has built a taxonomy of such verbs in relation to the REA model which is based on Speech act theory []. The intentional value of verbs is more fine-grained than the REA concepts identification-commitment-actualization. It is possible to use the verbs as an indication of the level of commitment.

An e-business system, as described before, can only allow utterances (being decisions and reports of events) that have previously been defined (‘typified’). Such system cannot enforce the parties to let certain events happen. The enforcement must be modeled in the process itself explicitly and may get the form of an allowed ‘reminder message’ from the other party after a time out. Or it may escalate beyond the agreed e-business process (e.g. the party may be sued to court).

REA adaptation

In order for the REA model to support e-business processes, we propose the following modifications to the REA model:

1. Widen the scope of the commitment concept to include other (e.g. legal) obligations.
2. Make commitments/obligations conditional (to other commitments or events) in order to be able to define the process flow.
3. Enrich commitments with verbs that express the level of commitment, and define communication patterns that fit those levels (e.g. propose-accept, request-response, commit-report).
4. Remove the direct duality association between economic events. Events are meaningful in relation to the commitment or obligation that prescribes them, not directly to other events.
5. Remove the direct reciprocity between commitments. Commitments have been agreed in the contract and the contract only defines the association between commitments and obligations.
6. Decompose the contract in multiple levels, where higher levels set conditions and boundaries for lower levels. High levels may not have been agreed bilaterally, but define contextual (e.g. legal) conditions binding both parties.

In figure 3 the adaptations are illustrated in a simplified REA model.

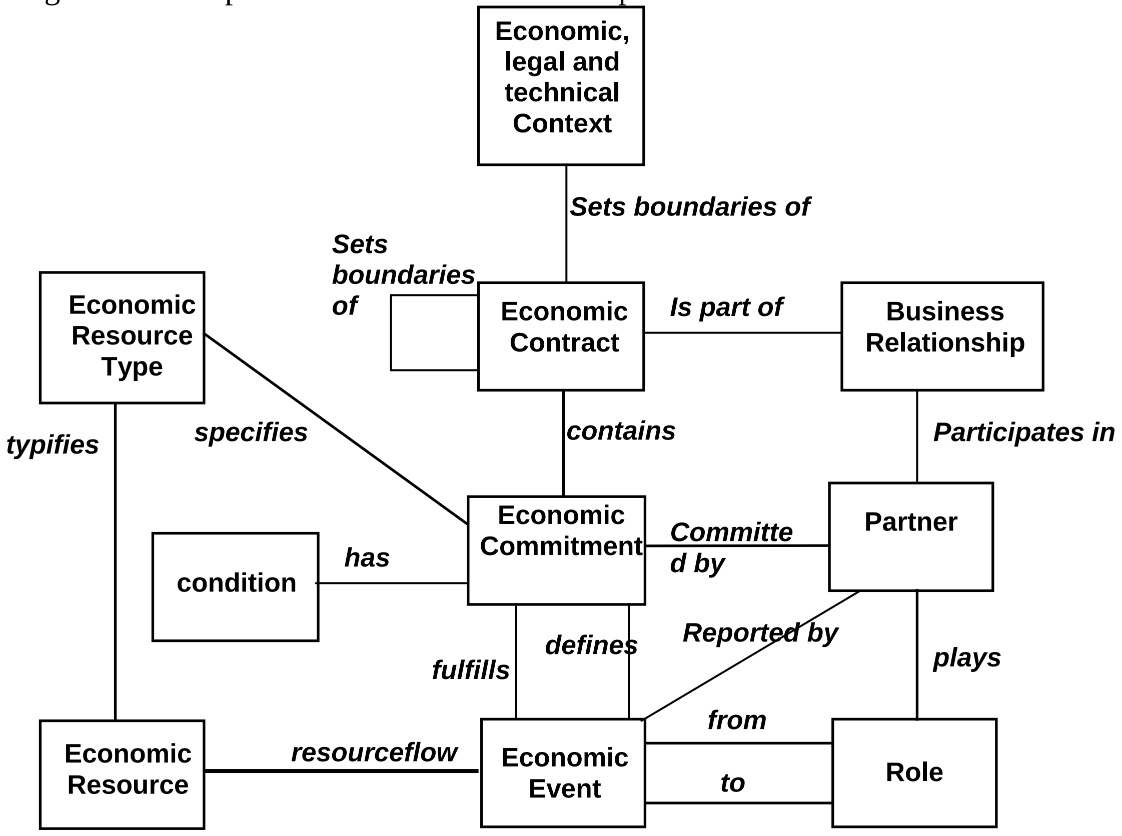


Figure 3. Simplified adapted REA model.

References

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

[2] UN/CEFACT' s Modelling Methodology, CEFACT/TMWG/N090R10, November 2001, available at www.untmg.org (accessed 21 May 2006).

[3] P. Jayaweera (2004) “A Unified Framework for e-Commerce Systems Development: Business Process Pattern Perspective Department of Computer and Systems Sciences” Stockholm University and Royal Institute of Technology, Stockholm, Sweden.

[4] Austin J. L., “How to do things with Words”, Cambridge, MA: Harvard University Press,2nd Edition 1975.

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

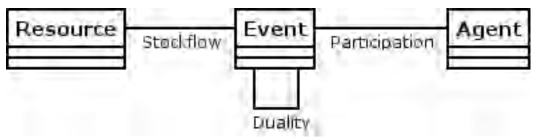


Fig. 1. The Resource-Event-Agent metamodel

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.

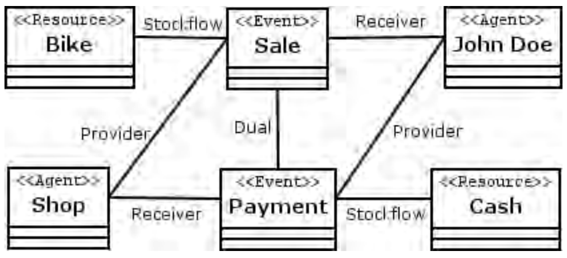


Fig. 2. The *transfer exchange*

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.

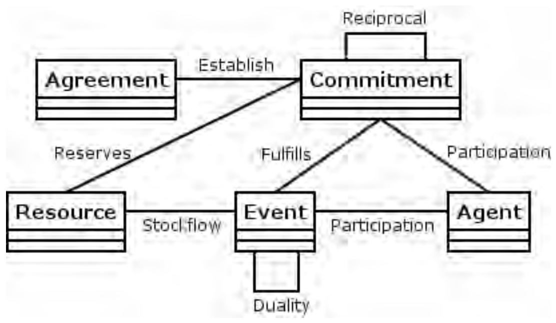


Fig. 3. Commitment and agreement

The agreement entity organizes pairs of commitments such that complex business transactions can be ordered in a hierachical 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 resourcetype. 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

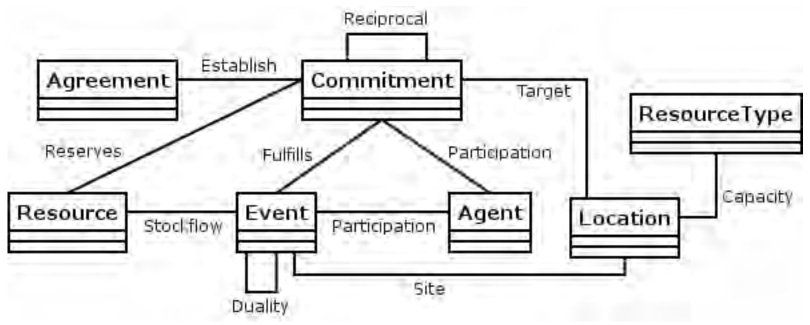


Fig. 4. Location

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 ambigous 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, manufactoring, 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**
Sale	yes	no	no
Payment	yes	no	no
Produce	yes	yes	yes
Use	no	no	no
Consume	yes	yes	yes
Load	no	no	yes
Unload	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 horisontal 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, wiz., 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 necesarily 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 bying 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 execption 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 coloumns 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, wiz., 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`.

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` (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. Addisson-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, Enshede, 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.

Revised Classification of the Enterprise Information Architecture Elements

Pavel Hruby, Jesper Kiehn
Microsoft Development Center Copenhagen, Frydenlunds Allé 6, 2950 Vedbaek, Denmark
phruby@acm.org, jkiehn@microsoft.com

Abstract

REA (resources, events, agents) is an emerging mainstream ontology for the enterprise information architectures. However, the REA ontology is underspecified, which opens for the possibility of various interpretations of the concepts and for various incompatible modeling styles. This paper shows that introducing the concepts of ownership and other rights into the REA ontology enables creation of semantically precise application models.

Keywords: *enterprise information systems; ontology; REA*

1. Introduction

REA (resources, events, agents) is an ontology for describing business systems. The core REA model describes a business system as a set of economic resources, economic events, and economic agents. *Economic agents* are individuals or organizations capable of having control over economic resources, and transferring or receiving the control to or from other individuals or organizations. *Economic resources* are things that are scarce, that have utility, and that users of business applications want to plan, monitor, and control. *Economic events* represent either an increment or a decrement in the value of economic resources. The core REA model is illustrated in Figure 1.

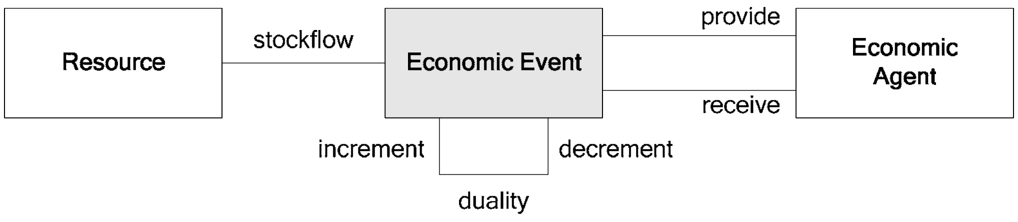


Figure 1. Core REA model (adapted from Hruby 2006)

Understanding REA fundamentally enhances the potential of application designers to configure business solutions without omissions. REA defines a set of domain rules (axioms) that assure consistency of the enterprise information architecture:

- At least one increment event and one decrement event exist for each economic resource. Likewise, every economic event must be related to an economic resource.
- Every increment economic event must eventually be related by duality relationship to a decrement economic event, and vice versa.
- Every economic event must be related to both provider and recipient economic agents.

REA was originally proposed as a generalized accounting model (McCarthy, 1982). Since then, it has been extended by McCarthy and Guido Geerts to a framework for enterprise information architectures and ontology for business systems (Geerts GL, McCarthy WE 2000a, 2002). REA became the foundation for several electronic interchange standards, such as ebXML and Open-edi.

The objective of this paper is to show that introducing the concept of rights, and especially the ownership rights into the semantics of the REA stockflows leads to precise modeling rules for REA application models.

The rest of this paper is structured as follows. The next section illustrates several REA modeling problems, which originate in the underspecification of the REA ontological categories. The next two sections discuss the original and the revised categories of stockflow relationships with the concept of rights. The last two sections illustrate how the concept of rights influences other REA modeling elements than stockflow, and how it helps to solve the REA modeling problems illustrated in the section 2.

2. REA Modeling Problems

REA as an ontology is underspecified, which (a) makes REA hard to understand, as some of the modeling questions are not answered by the ontology and are open for free interpretation, (b) allows for creating models that formally satisfy the REA axioms, but are incorrect from the business perspective. As an example, we'll describe several problems, which we repeatedly came across when trying to apply REA, and when explaining it to people with software background.

2.1 Are Economic Events Moments or Time Intervals?

Are the REA economic events instantaneous or do they last over a period of time? Answer to this question has the fundamental impact on the REA models such as rental; e.g. should a rental of a car be modeled as a single economic event with the start and end in different points in time, or as two instantaneous economic events: the start of the rental, and the end of the rental? Likewise, should a transport of an item be modeled as a single economic event with the start and the end in different points in time, or as two instantaneous events: the load and the unload event? The current REA ontology does not give an answer, as both options satisfy the REA axioms. However, this ambiguity contributes to possible misinterpretation of the REA models.

2.2 What Does "Produce" Mean?

Does the REA concept of "produce" (Geerts GL, McCarthy WE 2000a) mean "create" a resource, or can it also mean "modify" a resource? E.g. should transport (changing a location of a resource) be modeled as a consumption of the resource in the original location and a creation of a new resource in the destination? Or, should it rather be modeled as modifying a location property a resource but not its existence? Likewise, should "TV Marketing" be modeled as a consumption of the original resource and producing a new resource with a new feature called "Known from TV"? Or, should it rather be modeled as an event modifying the "Known from TV" feature, but not changing the existence of the resource? Both options are valid with respect to the REA axioms, but the semantic of the models is different, which opens space for ambiguity.

2.3 What Does "Inside" and "Outside" Agents Mean?

The concepts of inside and outside agents were well-defined in the original REA paper (McCarthy 1982), but with the introduction of the conversion processes into REA (Geerts, McCarthy 2000a), the meaning of the inside and outside agents became unclear, because the economic events in the conversion processes no longer transfer the resources between the inside and outside agents.

3. Classification of REA Stockflows

Stockflow is a relationship between an economic event and an economic resource, see Figure 1. There are several categories of stockflow relationships in REA. The *Inflow* and *Outflow* are the categories of stockflows in the original REA model (McCarthy 1982), which focused on exchange processes (transfers of economic resources). The *Produce*, *Use* and *Consume* are the categories of stockflows in the REA model describing conversion processes (transformations of resources) (Geerts, McCarthy 2000a). This classification scheme is outlined in Table 1.

Event Category	Stockflows in Conversion Process	Stockflows in Exchange Process
Increment	*Produce*	*Inflow*
Decrement	*Use* *Consume*	*Outflow*

Table 1. Classification of REA stockflow relationships

In this paper we use the terms Use and Consume as they are used in English: a resource ceases to exist after the Consume event, and still exists after the use event (i.e. what is consumed is eaten). The Produce in Table 1 means both create a new resource and modify an existing resource. The Inflow, and Outflow mean transfer of some rights to a resource from the provider to the recipient economic agent.

4. Revised Classification of REA Stockflows

In this section we'll show that adding a new criterion to the classification of stockflow relationship resolves the problems related to the ambiguity mentioned in section 2. The new criterion classifies stockflow relationships according to the rights associated with the economic resource that are influenced by the related economic event. The new classification scheme is outlined in Table 2.

One category of stockflows changes (i.e. creates, transfers or destroys) ownership rights. An enterprise establishes the ownership rights to a resource either by taking (buying) it or by creating it. An enterprise loses the ownership rights to a resource either by giving (selling) it or by consuming it. Therefore, the new categories of stockflows are Create and Take (related to the increment events), and Consume and Give (related to the decrement events).

The other category encompasses the stockflows that do not influence the ownership rights. The stockflows Produce (now with the precise meaning of modifying an existing resource) and Use do not change any rights. The stockflows Borrow and Lend do not change ownership, but transfer other rights, such as the right to use the resource. The terms Take and Give should be understood as generalized terms for transferring some rights to a resource, but not the ownership right.

Event	**Ownership**	**Stockflows in Conversion Process**	**Stockflows in Exchange Process**
Increment	*Changed*	*Create*	*Take*
Increment	*Unchanged*	*Produce*	*Borrow*
Decrement	*Changed*	*Consume*	*Give*
Decrement	*Unchanged*	*Use*	*Lend*

Table 2. Revised classification of REA stockflow relationships

The original Inflow relationship is now represented by the Take and Borrow, and the original Outflow by the Give and Lend relationships. Take and Give transfer ownership, Borrow and Lend transfer all other rights except of ownership.

The original Produce relationship is now represented by Create, which creates a new resource, and Produce with redefined semantics to mean the change some features of an existing resource. The semantics of Use and

Consume is unchanged (Consume destroys the resource, while after Use the resource still exists).

The stockflows Create, Take, Produce and Borrow are related to the increment economic events, the stockflows Consume, Give, Use and Lend are related to the decrement events.

5. Revised Semantics of the REA Elements

The classification scheme for stockflow also determines the semantics of the economic events and agents. The new semantics of economic events is illustrated in Table 3.

Ownership	Events in Conversion Process	Events in Exchange Process
Changed	*Create or terminate ownership* *Change resource existence* *Last over a period of time*	*Transfer ownership* *Do not affect resource features* *Are instantaneous or last over period of time*
Unchanged	*Do not influence rights* *Do not change existence* *Last over a period of time*	*Transfer rights but not ownership* *Do not affect resource features* *Last over a period of time*

Table 3. Revised semantics of economic events

The evens related to Take and Give stockflows might be instantaneous or last over period of time. All other events last over a period of time.

The revised classification scheme does not require the concepts of the increment and decrement events, and therefore can be used both in the REA models in the trading partner view, and the independent view. Distinguishing between an increment and a decrement is necessary in the trading partner view. In the independent view, the concepts of increment and decrement are redundant, therefore, the Take and Give will be replaced by the Transfer Ownership event, and the Borrow and Lend will be replaced by the Transfer Rights event.

The concept of the *Participation* relationship (Geerts, McCarthy 2000a, 2000b, 2002) between an economic event and an agent can be replaced by the *Provide* relationship and the *Receive* relationship (Hruby 2006), see also Figure 1. These relationships have different semantics in the exchange and in conversion processes. For the lack of better terminology we call the *Participation* in the exchange process the *Provide Rights* and *Receive Rights* relationships; and the *Participation* in the conversion the *Provide Control* and *Receive Control* relationships.

The *Provide Rights* and the *Receive Rights* relationships link the economic agents, who are legal entities capable of holding ownership rights to economic resources, and transferring them to other agents, with the events *Take*, *Give*, *Borrow* and *Lend*. The *Provide Control* and the *Receive Control* relationships link the economic agents, which are actual people capable of controlling economic resources, but not having ownership rights to them, with the economic events *Create*, *Consume*, *Produce* and *Use*.

There is still a single concept for the *Economic Resource*, because an economic resource can be related to every stockflow in the scheme. However, the resources expose a different interface to the conversion and to the exchange stockflows (and consequently, to the economic events). The stockflows in the conversion processes view a resource as a portfolio of features. The stockflows in the exchange process view a resource as a portfolio of rights.

The new classification scheme allows for more precise formulation of the domain rules (axioms) as follows.

- Every economic resource must be related by a Create or Take relationship to an increment economic event, and by a Consume or Give relationship to a decrement economic event
- Every economic event must be related to an economic resource.
- Every economic event with the Create or Produce stockflow must be related via a Conversion Duality to an economic event related to the Use or Consume stockflow.
- Every economic event with the Take or Borrow stockflow must be related via an Exchange Duality to an economic event related to the Give or Lend stockflow.
- Every economic event with the Create, Consume, Produce and Use stockflow must be related by a Provide Control and Receive Control relationships to economic agents.
- Every economic event with the Take, Give, Borrow and Lend stockflow must be related by a Provide Rights and Receive Rights relationships to economic agents, which represent parties with different economic interests.

6. Modeling Problems Resolved

The new classification scheme gives a more specific answer on which economic events are instantaneous and which last over a period or time. Therefore, a rent of a car will be modeled as a Borrow or Lend, depending on whether the model represents the renter’s or owner’s viewpoint. Therefore, the rent of a car is an economic event that lasts over a period of time. The transport of an item will be modeled as a Produce relationship, and the economic event lasts over a period of time as well.

The problem of the inside and outside agents is solved by removing these concepts from the ontology (there is now just one category of economic agents), and by refining the concept of the Participation relationship into the Provide Rights, Receive Rights, Provide Control and Receive Control relationships.

Representing the Produce stockflow either as the Create and or the Produce relationships solves the problem of modeling the marketing process, The “TV Marketing” can now be modeled by using the Produce relationship (with the meaning of “modify”) between an increment event and the resource, as the redefined Produce relationship (with the meaning of “modify”) does not change the existence of the resource.

7. Conclusions

The revised classification scheme of the enterprise information architecture elements can be seen as a refinement of the REA ontology, while preserving its fundamental principles and concepts. The revised scheme focuses on the operational semantics of REA, but it can consistently be extended to cover other REA concepts, such as commitments and contracts. Authors would appreciate feedback, comments and suggestions.

References

Dunn CL, Cherrington OJ, Hollander AS (2004) Enterprise Information Systems: A Pattern Based Approach, McGraw-Hill/Irwin, New York

Geerts GL, McCarthy WE (2000a) The Ontological Foundations of REA Enterprise Information Systems. Paper presented at the Annual Meeting of the American Accounting Association, Philadelphia, PA.

Geerts GL, McCarthy WE (2000b) Augmented Intensional Reasoning in Knowledge-Based Accounting Systems. Journal of Information Systems, Volume 14, No. 2, 2000, pp. 127-150.

Geerts GL, McCarthy WE (2002) An Ontological Analysis of the Primitives of the Extended REA Enterprise Information Architecture” at http://www.msu.edu/user/mccarth4/

Haugen, B (2005) Resources and Rights, Discussion in REA Technology Mailing List, http://groups.yahoo.com/group/REATechnology

Hessellund A (2006) Supply Chain Modeling with REA, IT University Copenhagen, at http://www1.itu.dk/sw43956.asp

Balthazar S, Chohan A, Hessellund A (2005) Supply Chain Integration (in Danish), MSc. Thesis, supervised by Østerbye K, IT University Copenhagen, Denmark

Hruby P, Kiehn J, Scheller KV (2006) Model-Driven Design Using Business Patterns, Springer Verlag

Jaquet M (2003) Realistic – A REA Model without Perspectives (In Danish). MSc. Thesis, IT University Copenhagen

McCarthy WE (1982) The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment. The Accounting Re-view (July 1982) pp. 554-78

Sowa JF (1999) Knowledge Representation: Logical, Philosophical, and Computational Foundations, Course Technology