

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 '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 Pacioli¹ 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 5 for 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).

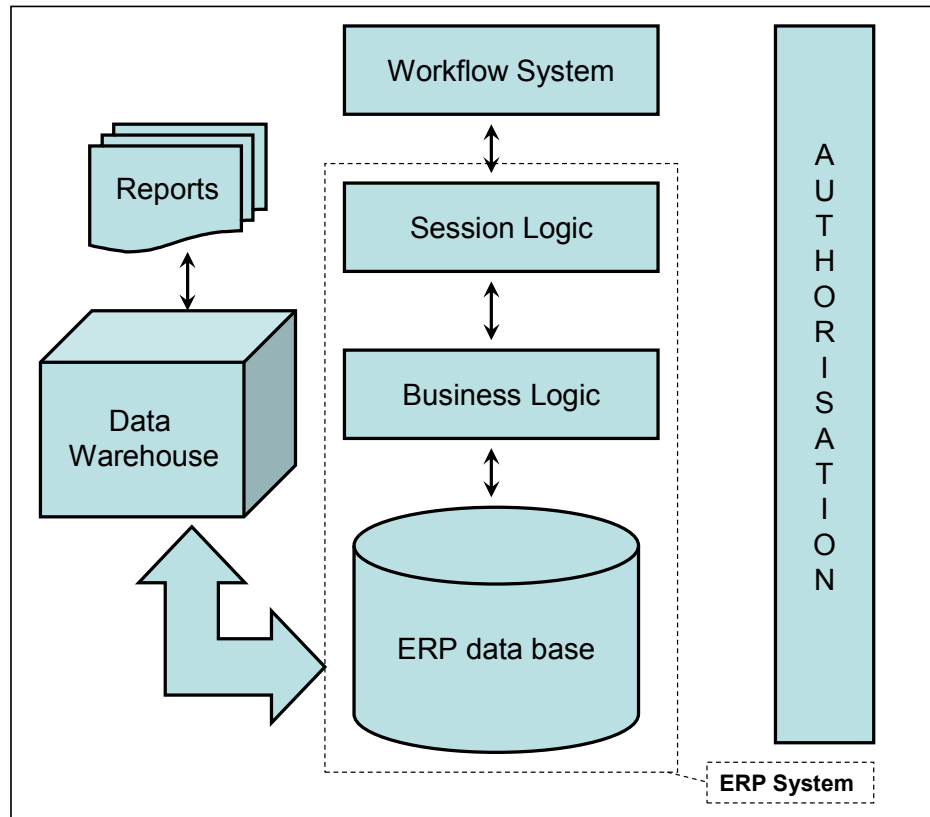


Figure 1: Main components of a Business Information System

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 ERP 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 Vandebossche (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 started 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 with 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 it 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 and 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 into 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 in 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 in 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 basic 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 ‘extendibility’. 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.

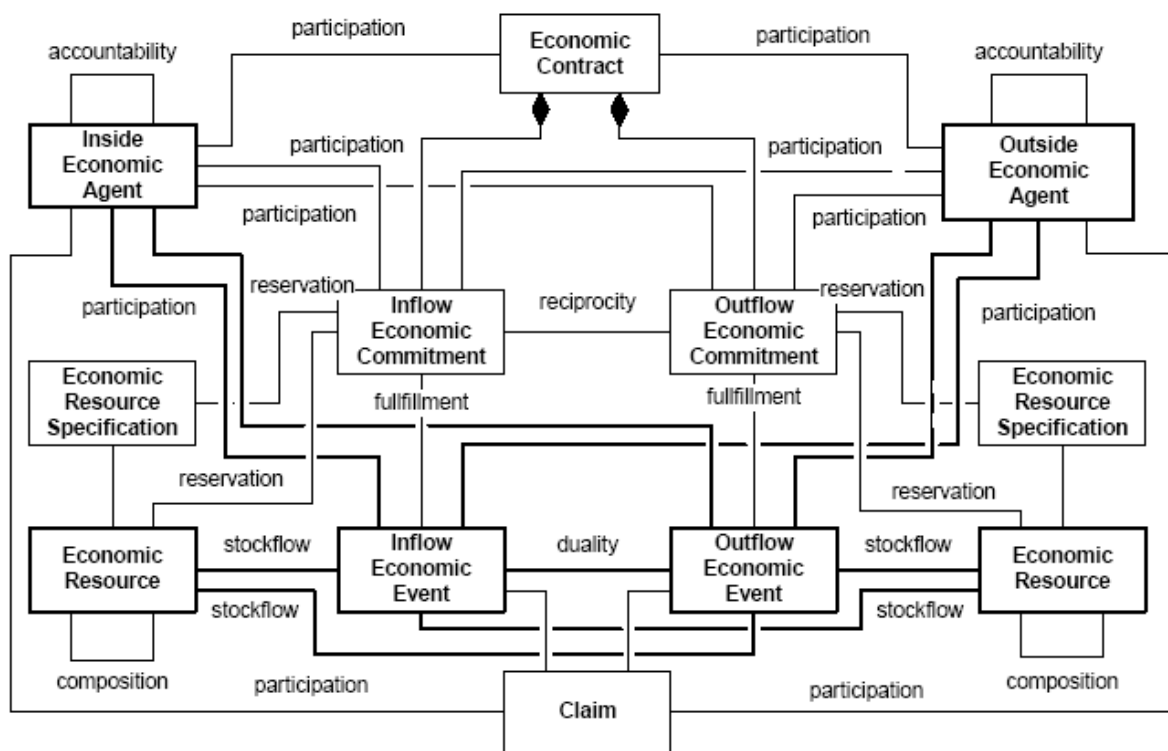


Figure 2: Object version of the extended REA Model (Source: Hruby, 2004)

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 languages² 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 will 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 too 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. McCarthy (1997). The REA Accounting Model: Intellectual Heritage 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 Association 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). *Preziale Wirtschaftlenkung, Bd 2: Preziale 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 Preispolitik*, 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.