Congruent and Meronymic Constellations in the REA Ontology

William E. McCarthy, Michigan State University
Guido L. Geerts, The University of Delaware
Graham Gal, The University of Massachusetts

The REA metamodel has been expanded in recent years beyond the basic accountability pattern that was first specified in *The Accounting Review* in 1982 by McCarthy. These expansions are explained in detail in a series of publications by Geerts and McCarthy (2000, 2006) and in the international ISO standard for an accounting and economic ontology (ISO, 2015). This expanded metamodel is illustrated in Figure 1, and the results are an expansion from just the accountability layer of a business transaction (“what has occurred”) to an additional scheduling layer (“what has been specified or reserved”) and an additional policy layer (“what could be or should be”). As illustrated in Figure 2, the scheduling layer extends REA specification from the past and near present to the planned future; while the policy layer provides business rules and constraints on both present and future behavior.¹

The idea that the REA metamodel is actually a “pattern” for business process analysis was explained by McCarthy (2003,430):

> In a semantic database design (and also in its closely related analysis cousin of object-oriented design), the hardest step is always the first: coming up with a good list of candidate entities (or objects or classes) on which to base the rest of the analysis. … To overcome this difficulty, the analysis patterns movement was born (Fowler 1997; Hay 1996; Coad 1995) in the early 1990s. The REA accounting model preceded this work by a decade, but its basic framework of interlocking constellations of Economic Resources, Economic Events, and Economic Agents was actually a complex aggregation of some of those patterns that surfaced in the nineties.

Most basically, a pattern is a stereotypical constellation of classes that gives a designer an idea of what objects or categories to look for when analyzing a particular piece of

¹ Most of the text in this article and all of its embedded figures have been adopted from a research monograph in progress by William McCarthy, Guido Geerts and Graham Gal (2016): *The REA Accounting Model as an Enterprise Ontology*. 
reality. The AI analog of an analysis pattern is the concept of a *frame* (Minsky 1975) where components of the pattern are called *slots*. McCarthy (1987) used this strong correspondence to illustrate representation of the initial REA template as an economic event frame with slots for inside agent, outside agent, and resource. Applied to the REA metamodel of Figure 1, we could say (in artificial intelligence terms) that we have a frame for a business process with slots for components such as resources, events, agents, commitments, types, etc.

In the rest of article, we will discuss situations where components of the REA metamodel (or slots in the REA frame) are either expanded or contracted systematically in use, primarily because of representation constraints but also because of business or industry practice. It is important to understand how these expansions and/or contractions work systematically for two reasons:

1. so they can be accommodated or even reversed in the context of using REA for automated or intensional reasoning (Geerts and McCarthy 2000), one of the prime functions for a well-developed domain ontology; and
2. so accounting system users and analysts can understand how full-REA systems must sometimes be reduced or expanded to meet system implementation constraints.

We attack the more complex issue of REA slot contraction first.

**Contraction of metamodel components – Conceptual congruency**

“To be is to be the value of quantified variable.” This is a famous slogan of the philosopher Willard Van Orman Quine (1992) as quoted and explained by John Sowa (2000 52), a slogan to be used as a criterion for admission of an ontological category into a representation scheme. In earlier work, Quine (1969, 23), had expressed this notion as “no entity without identity.” In both philosophical cases, the import of the ideas is similar: it is difficult to represent and discuss a concept without providing a name for identifying the different individual instances of that concept.

For REA implementation, “no entity without identity” means that a candidate class for a particular model (at MOF Level-0) must have an identifying attribute that can stand for the instances of that class. For a database, this means that all classes must have a primary key (PK) attribute (or combination of attributes) that is both universal (every instance possesses that attribute) and unique (every instance’s value for that attribute is different from all other instance values). Although it is sometimes possible that database and object-oriented (OO) implementation have system-generated primary keys, it is always the case that there is domain relevant name for individual objects that can act as an identifier. For example, in an OO language like JAVA, instances or objects in the class CUSTOMER need differentiated names like firstCustomer, secondCustomer, etc.

The requirement for an identifier is thus a representation constraint on an REA implementation. However, a notion called *conceptual congruency* mitigates the effect of this constraint on the process of developing an REA-compliant implementation. The
idea of conceptual congruency was discussed in several places in the original REA paper, but we explain it more completely here with multiple examples.

In Figure 3, we illustrate generically how conceptual congruency works. Two classes and the association between them are simply combined into one class, because the set of instances postulated for each are deemed to be (eventually) congruent. The one class can then fill two slots in the REA frame with the list of attributes for those classes describing characteristics of both components. For example, a reservation for a hotel could have “reservation#” as an identifier, and the same class could actually represent both the commitment for a stay in the hotel and the actual stay itself once a customer arrives. The congruent class could have two attributes (among others) describing both slots, like “planned number of days for the stay” and “actual number of days for the stay.” The primary reason for folding two classes together is the difficulty in finding an identifier for both; however, it is also true that industry custom or practice sometimes dictates combination.

In Figure 4, we have four common examples of simple REA conceptual congruency.

- In the northwest corner, we have the case just mentioned where a single class fills the slot for both a commitment and an economic event. Besides the hotel stay example, other common examples include cases where a “purchase-order#” identifies both a purchase commitment and an actual purchase or where a “production-order#” identifies both a planned production run and an actual production run. Both of these cases are common in industry practice, but newer technology – like using timestamps here for the actual event instead of a document number—could undo the need for congruent classes in such cases.

- In the northeast corner, we have the case of “nail” resources mentioned in chapter 3. This is a circumstance almost always brought about by representation constraints (i.e., identifier difficulties), although RFID (radio frequency identification) technology is clearly moving the identification possibilities frontier closer to obviating the need for congruency in many industries.

- In the southwest corner, we have examples where a large-grained increment event (like a consulting job or an advertising campaign) becomes upon its completion the actual resource to be exchanged or consumed in a downstream revenue or production process. For example, completed audit engagements in effect become the economic resources that in-charge audit partners sell to their clients.

- In the southeast corner, we have a congruency example common to simple cash exchanges, like a vendor selling a hot dog to a customer at a ball game. In this case, the increment and decrement events occur almost simultaneously, so treating them as one occurrence makes representation sense.

In Figure 5, we portray more complicated examples of conceptual congruency, both of them dealing with duality associations:
In the congruency example (surrounded by dotted lines) on the left, we have three decrement economic events being combined that represent outflows of three different economic resources. Such a combination is often necessitated by difficulty in finding an identifier for multiple outflows. For example, we could have a sale that uses inventory being combined with the truck being used to deliver the sale and with the labor of the truck driver doing the delivery. In this case we might have a single identifier (like a “timestamp” or an “invoice-number”) for a class that represents all three with example attributes like “quantity of inventory sold,” “miles driven by the truck,” and “time spent by the driver.”

In the congruency example on the upper right, we have the quite common case of an increment event (like a cash receipt) being linked to an inflow resource (like cash), but which also uses some labor (like the time of the cashier) doing the event processing. Similar to above, we might have a single identifier (like a “remittance-advice-number” or a “timestamp”) for the class with example attributes like “cash receipt amount” and “time spent by cashier.”

McCarthy, Geerts, and Gal (2016) illustrate a number of other important accounting uses for slot contraction with examples including the acquisition/consumption of services and the tracking of labor costs in a business process. However, we move now to the opposite idea of slot expansion.

Expansion of metamodel components – Meronymic classes

Again, the REA metamodel of Figure 1 illustrates a business process frame with the individual classes representing slots in that frame. We demonstrated above how the notion of conceptual congruency can contract those slots. In this section, we show how those slots can be expanded. Such expansion is quite common when full design of a particular data representation needs to become more detailed. Basically, we treat the expansion as a whole-part constellation: an abstraction termed “meronymic inclusion” by Storey (1993, p.463) who notes that the term derives from the Greek word “meros” for part. Storey also details seven different types of meronymic relationship, but we will concentrate here on just the higher level concept. More detailed analysis of meronymic expansion can be found in Motschnig-Pitrik (1993).

Figure 6 illustrates the basic idea of meronymic expansion – a representation of an example class (the whole) is expanded to include some of its component parts (the meronym). Examples of meronymic relationships include these:

- A finger is a meronym of a hand;
- An engine is a meronym of a car; and
- A song-performance is a meronym of a rock-n-roll concert.
Many REA components have meronymic structures, but we illustrate in detail just one here as part of an REA implementation (M1) for an example company – Alaskan Aircraft Expeditions (AAE) – which conducts tours with aircraft to various locations, so its revenue process is one where they use an aircraft and consume varying amounts of employee labor on expeditions. “Expedition” is thus the decrement event in the revenue business process. However, the whole expedition is broken into “expedition tickets” for the purpose of matching customers to a particular tour, and what they pay for is a ticket (obtained from a booking agent) that allows them to be part of the expedition. The expedition-ticket is a meronym of the expedition. In Figure 7, we illustrate this in more detail with a process level REA class diagram with the whole-part constellation again surrounded by a dotted line. Readers should note how expanding this economic event into parts allows better specification of the relationships connecting the event to its resources and agents.

Meronymic relationships are a subset of the more general “aggregation” abstraction pioneered by Smith and Smith (1977) and explained in an accounting context by McCarthy (1987). Aggregation also applies when a business process is expanded to multiple subordinate business processes, when a single business process is decomposed into business events, or when one business event is decomposed into multiple business events.

Summary

In the referenced monograph, McCarthy, Geerts, and Gal (2016) discuss and develop a wide range of issues involved in more fully developing the REA Enterprise Ontology. In this excerpted article, we present just one of those issues – how to expand and contract the slots in an REA business process frame.

References


International Standards Organization. 2015. ISO 15944-4 - Information Technology -
Business Operational View - Part 4: Business Transaction Scenarios -
Accounting and Economic Ontology. Geneva, CH: ISO.

McCarthy, W., Geerts, G., and Gal, G. 2016. The REA Accounting Model as an
Enterprise Ontology. East Lansing.


—. 2006. "Nantasket Casket, Basket, and Gasket teaching case." Semantic Modeling of
Accounting Phenomena (SMAP) Workshop.

Memorial Lecture Series, (The University of Missouri). (also published in Artificial
Intelligence in Accounting and Auditing:The Use of Expert Systems, M.

—. 2000. "REA Accounting Models – The Need for Both Internal & External Views in a
World of Interconnected Enterprises , March 2000." Invited Keynote Presentation
to European Conference on Accounting Information Systems., Munich Germany.

Accounting Systems in a Shared Data Environment." The Accounting Review 57
(3): 554-578.

McCarthy, William, Guido Geerts, Graham Gal.: 2016. The REA Accounting Model as
an Enterprise Ontology. Michigan State University.

Minsky, Marvin. 1975. "A Framework for Representing Knowledge". In The

Data/Knowledge Modelling." CAiSE '93 Proceedings of Advanced Information

University Press.


![REA Metamodel](source: McCarthy, Geerts, and Gal, 2016)
Figure 2 -- REA Extensions: Commitments and Policy-Level Specifications (source: McCarthy, Geerts, and Gal, 2016)

Figure 3 -- Conceptual congruency of two classes (source: McCarthy, Geerts, and Gal, 2016)
Figure 4 -- Conceptual congruency examples (M2)
(source: McCarthy, Geerts, and Gal, 2016)

Figure 5 -- Conceptual congruency of duality associations (M2)
(source: McCarthy, Geerts, and Gal, 2016)
Figure 6 – Meronymic Expansion
(source: McCarthy, Geerts, and Gal, 2016)

Figure 7 – Meronymic Expansion of an Event Slot (M1)
(source: McCarthy, Geerts, and Gal, 2016)