Resources, Events, Agents Pattern

(Position Paper for Workshop on Patterns for Software Architecture at OOPSLA 2002)

Pavel Hruby

Microsoft Business Solutions
Frydenlunds Allé 6
DK-2950 Vedbaek, Denmark
E-mail: phruby@microsoft.com

Abstract

Have you ever tried to describe an object model of a business system and struggled to find the right relationships between business entities, such as customers, business partners, products, sales and purchase orders, invoices and credit memos? Have you ever wanted to know a simple rule for modeling the business system in a consistent manner? The Resources-Events-Agents (REA) pattern is a fundamental architectural pattern for building extensible models of business systems. This pattern can be extended by a number of other patterns, such as commitment, claim, roles, due date, address, classification, and account, which altogether form the Universal Enterprise Model pattern language.

Context

Business is based on exchanges of economic resources. For example, a customer at a shop buys a product. The product is the resource and the sale is the outflow of resources. The payment for the product is an inflow or resources and the cash is the resource. If the deal is fair, both partners agree that the value of the exchanged resources is the same. That is, the agreed value of the product is the same as the amount of cash received in return.

Problem

How do we model exchanges of resources in business systems?

Forces

1. You want to model the rules that apply to *all* economic systems. However, most analysis patterns and data models are often domain dependent, because they reflect experience of specific software consultants.

2. You want to model things that are shareable and reusable across various application domains, and where details of specific applications are ignored. You could build your object model from user requirements, but it would be very difficult to find the right abstractions valid for *all* economic systems.

3. You want to build an extendable system, which can be extended by new concepts without changes to the foundations of the system. This is difficult, because you must sort out or generalize the domain specific knowledge.

4. You want well-defined modeling rules that enable you to ensure that your model is consistent. For example, how do you keep track of a party who gives resources away, and eventually receives other resources in return? Other examples of consistency questions include: if company sells a product, how does the company obtain it? Or what does a company get in return for providing a customer with certain benefits?

Solution

Consider the economic activities as a sequence of exchanges of resources – the process of giving up some resources to obtain others. The following entities model the exchanges of resources.

Economic event represents the interval in time, or a moment in time when economic exchange occurs. Further more, economic event keeps track of the *value* of exchanged resources. The *stock flow* is a class attribute with two possible values: *inflow* and *outflow*. Inflow represents an event when a party receives ownership of resources, for example, incoming payment. Outflow represents an event when a party yields ownership of resources, for example, a shipment of goods. The *time interval* specifies a moment or interval in time when the exchange occurred. Some exchanges occur instantaneously, such as sales of goods; some occur over interval of time, such as rentals or services. Note that the change of ownership (the economic event) sometimes occurs at a different time to that of the physical movement of goods.

Resource represents the subject of trade.

Party represents an economic unit, or legal entity capable of exchanging economic resources with other parties. McCarthy and Geerts in their papers, for example [3], use the term *agent*. I use the term party, as *agent* in the software context has already other meaning.

Duality is a relationship between inflow economic events and outflow economic events. When the deal is closed and fulfilled, the values of inflow and outflow economic events are the same. It is *duality's* responsibility to keep track of the balance between the outflow and inflow. Duality is a many-to-many relationship. For example, several sales can be paid by one check, and one sale can be paid by several installments.

The model is illustrated in Fig. 1.

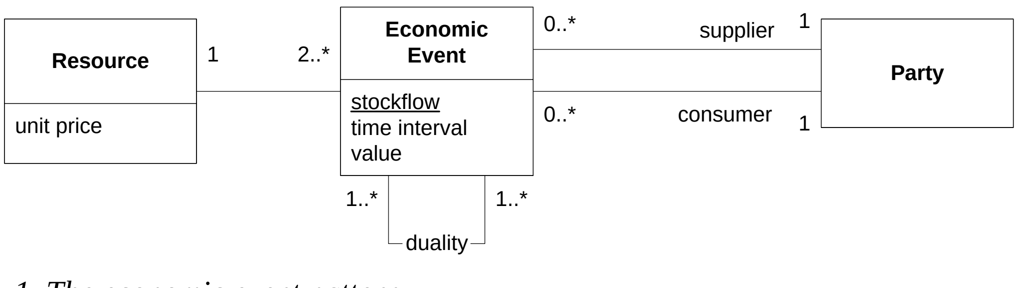


Fig. 1. The economic event pattern

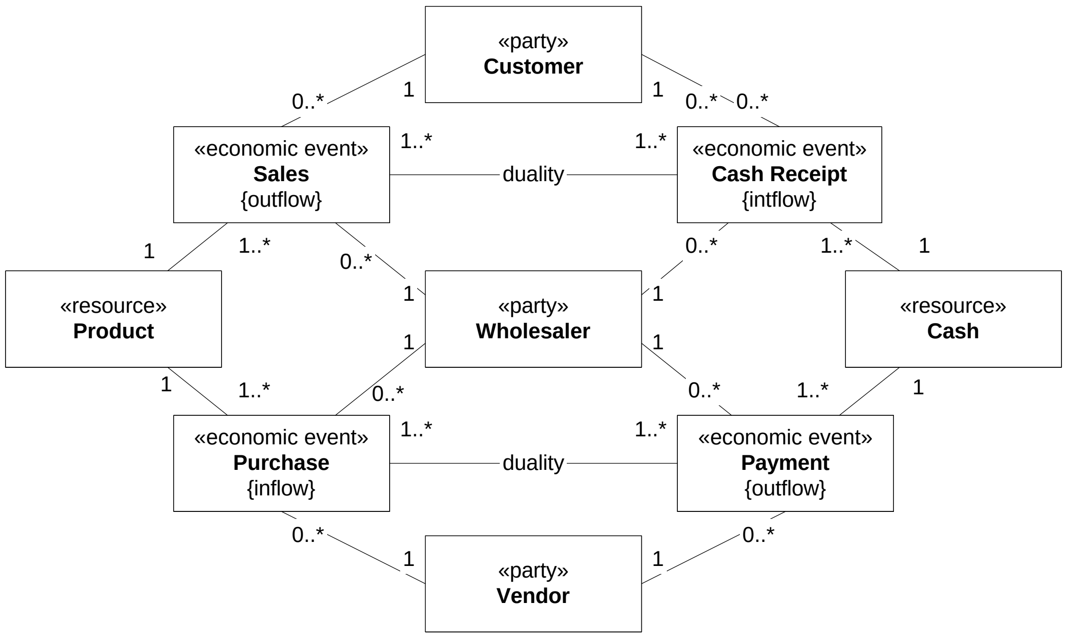
The following three rules apply for the model. They can be used to ensure consistency of a specific instantiation of this pattern.

1. At least one inflow and one outflow economic event exist for each resource. Conversely, each inflow and outflow economic event must be related to a resource. For example, goods related to the sales event must also be related to the purchase or production event (or some other event specifying how those goods are going to be obtained).

2. Each outflow economic event must have a duality relationship to an inflow economic event, and vice versa. For example, shipment to a customer (outflow) must be related to a customer payment (inflow).

3. Each economic event must have a relationship to two parties participating in the exchange. For example, each economic event must be related to the customer and vendor, employer and employee, and so on.

An example of a simple system is illustrated in Fig. 2. In this example, a wholesaler buys products from vendors and sells them onto customers. The *Customer*, *Vendor* and *Wholesaler* are the parties; the *Product* and *Cash* are the resources; *Purchase*, *Payment*, *Sales* and *Cash Receipt* are the economic events.



«party» Customer
«economic event» Sales {outflow}
duality
«economic event» Cash Receipt {intflow}
«resource» Product
«party» Wholesaler
«resource» Cash
«economic event» Purchase {inflow}
«economic event» Payment {outflow}
«party» Vendor
0..*
1
1..*

Fig. 2. Economic events for a wholesale system

Known uses

Cash sale. Product and cash are the resources; delivery and payment are the economic events.

Rental. The premises is the resource; yielding the right to occupy to the tenant is an outflow of resources. Payment of rent is an inflow of resources. Rental is an example of economic event that occurs over an interval of time.

Employment. From employer's perspective, an employee's time is the inflow of resources, salary is the outflow. Employment is another example when an economic event occurs over an interval of time.

Resulting Context

A software system based on this pattern registers economic events with all relevant data describing the state of the enterprise. The accounting artifacts, such as accounts, journals and ledgers, debits and credits can be reconstructed from these data. As a result, a REA-based system provides for more complete information about the enterprise than double-entry bookkeeping.

This pattern defines rules valid for *all* economic systems that exchange resources. This pattern forces developer or analyst to make a *complete* model of the economic exchanges, and think about non-obvious questions like "what resources do I get in return for paying my taxes?" This is often useful, but sometimes getting a complete and economically correct model is not easy.

Universal Enterprise Model

The REA pattern is the fundamental structural pattern, which forms the skeleton of the Universal Enterprise Model pattern language. The COMMITMENT pattern deals with contracts – agreements on economic events. The CLAIM pattern describes unbalanced exchanges of resources, for example, goods that has been shipped and not fully paid. The ACCOUNT pattern describes balances of resources and parties. The DUE DATE pattern describes when economic events are scheduled to occur. The ROLES pattern describes situations when a customer, vendor or employee can be the same physical entity. The CLASSIFICATION pattern describes types of parties and resources that are handled in a uniform way. The pattern map is illustrated in Fig. 3.

This pattern language has been described in [5].

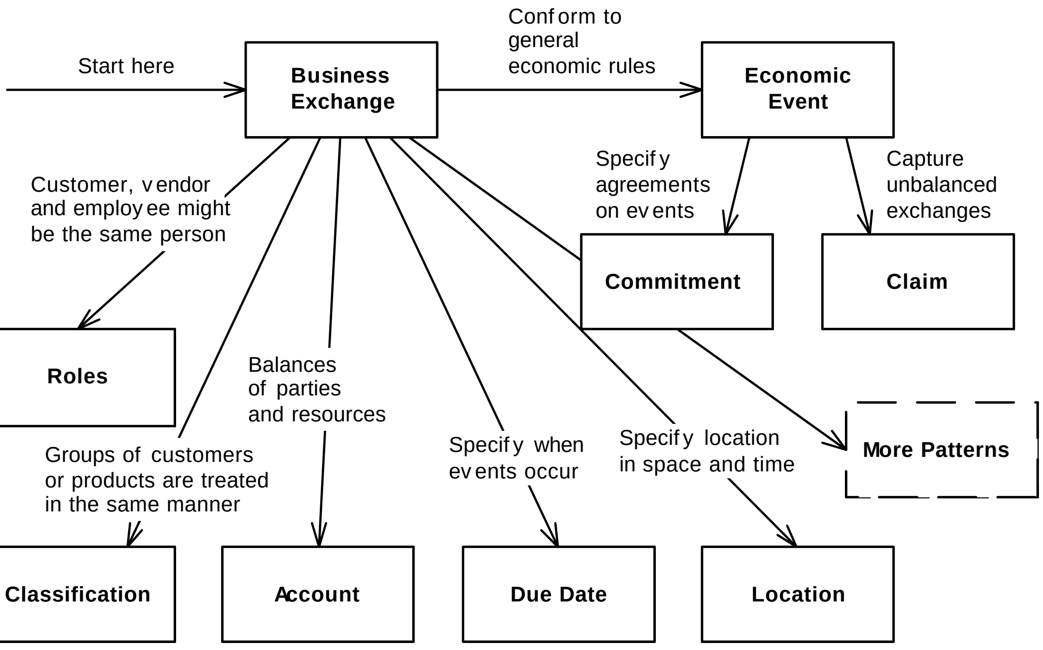

Start here
Business Exchange
Conform to general economic rules
Economic Event
Customer, vendor and employee might be the same person
Specify agreements on events
Capture unbalanced exchanges
Commitment
Claim
Roles
Balances of parties and resources
Groups of customers or products are treated in the same manner
Specify when events occur
Specify location in space and time
More Patterns
Classification
Account
Due Date
Location

Fig. 3. Universal Enterprise Model Pattern Map

References

[1] Coad, P., Lefebre, E., DeLuca, J.: Java Modeling in Color with UML, Prentice Hall PTR, 1999.
[2] David, J. S.: Three events that defined an REA methodology for systems analysis, design and implementation
[3] Guido L. Geerts and William E. McCarthy: The Ontological foundation of REA Enterprise Information Systems, 1999-2000.
[4] Hay, D.: Data Model Patterns, Conventions of Thought, Dorset House Publishing, 1996.
[5] Hruby: Business Relationships, The First Nordic Conference on Pattern Languages of Programs, draft available at http://www.plop.dk/vikinplop
[6] W E. McCarthy: The REA Accounting model: A generalized framework for accounting systems in a shared data environment. The accounting review (July) pp. 554-578, 1982.
[7] Silverston, L., Inmon, W. H., Graziano, K,: The data model resource book, John Wiley & Sons, Inc. 1977.