Defining Business Rules ~ What Are They Really?

the Business Rules Group

formerly, known as

the GUIDE Business Rules Project

Final Report

revision 1.3

July, 2000

Prepared by:

David Hay
Group R, Inc.

Keri Anderson Healy
Model Systems Consultants, Inc.

Project Manager:

Allan Kolber
Butler Technology Solutions, Inc.

Example by:

John Hall
Model Systems, Ltd.

Project team:

Charles Bachman
Bachman Information Systems, Inc.

Joseph Breal
IBM

Brian Carroll
Intersolv

E.F. Codd
E.F. Codd & Associates

Michael Eulenberg
Owl Mountain

James Funk
S.C. Johnson & Son, Inc.

J. Carlos Goti
IBM

Gavin Gray
Coldwell Banker

John Hall
Model Systems, Ltd.

Terry Halpin
Asymetrix Corporation

David Hay
Group R, Inc.

John Healy
The Automated Reasoning Corporation

Keri Anderson Healy
Model Systems Consultants, Inc.

Allan Kolber
Butler Technology Solutions, Inc.

David McBride
Viasoft

Richard McKee
The Travelers

Terry Moriarty
Spectrum Technologies Group, Inc.

Linda Nadeau
A.D. Experts

Bonnie O'Neil
MIACO Corporation

Stephanie Quarles
Galaxy Technology Corporation

Jerry Rosenbaum
USF&G

Stuart Rosenthal
Dyna Systems

Ronald Ross
Ronald G. Ross Associates

Warren Selkow
CGI/IBM

Dan Tasker
Dan Tasker

Barbara von Halle
Spectrum Technologies Group, Inc.

John Zachman
Zachman International, Inc.

Dick Zakrzewski
Northwestern Mutual Life

GUIDE Business Rules Project
Final Report

Table of Contents

GUIDE Business Rules Project
Final Report

Table of Contents (cont.)

Table of Figures

1
Introduction

The GUIDE Business Rules Project was organized in November 1993 to formalize an approach for identifying and articulating the rules which define the structure and control the operation of an enterprise. Systems analysts have long been able to describe an enterprise in terms of the structure of the data that enterprise uses and the organization of the functions it performs, but they have tended to neglect the constraints under which the enterprise operates. Frequently these are not articulated until it is time to convert the constraints into program code. While rules which are represented by the structure and functions of an enterprise have been documented to a degree, others have not been articulated as well, if at all. The Business Rules Project proposes to do this and, in doing so, to fill in the gaps for the kinds of business rules that have not been adequately documented in the past.

It is hoped that, as the task of defining business rules is better understood, techniques and tools will be developed to support the missing elements of the task. Techniques would include formal methods for describing rules rigorously, with tools translating these formalisms directly into program code or other implementation constructs. While graphic techniques have existed for some time to describe data structure and functions, formal methods for describing rules are relatively new and still being refined.

PROJECT SCOPE AND OBJECTIVES

The GUIDE Business Rules Project has been organized with four specific purposes:

- To define and describe business rules and associated concepts, thereby enabling determination of what is, and is not, a business rule.
- To define a conceptual model of business rules in order to express (in terms meaningful to information technology professionals) just what a business rule is and how it applies to information systems.
- To provide a rigorous basis for reverse engineering business rules from existing systems.
- To provide a rigorous basis for engineering new systems based on formal definitions of business rules.

The first two objectives have been met and the results are presented in this document. The Project articulates what constitutes a business rule and what about the rule should be captured and described. The discussions which led to this point have provided the groundwork for the latter two objectives, but much work remains to be done in these areas.

OVERVIEW OF THE PAPER

The first part of this document (Chapters 1-3) describes business rules in general ~ why we are concerned with them, how they are created, and what it means to formalize them. The second part (Chapters 4-6) describes in detail the conceptual model which the Business Rules Project developed to describe specifically what constitutes a business rule and what kinds of business rules exist. Portions of the model are shown with each explanation, and the entire model may be seen in Appendix A.

The model is drawn according to the conventions of IDEF1X, with a modification to the convention to make the relationship names easier to include in this text. The conventions for reading the model are contained in Appendix B of this document.

Throughout the paper, examples are drawn from a case study about a car rental company, EU-Rent. This case study was developed by Model Systems, Ltd. of the United Kingdom. The full case study is described in Appendix D.

THE RATIONALE

The techniques of systems analysis have evolved over the years to provide methods for describing many aspects of a business or government agency. We can now draw pictures of the way information flows through an organization, the sequence of actions an organization will follow, the structure of its operating information, and so forth. In some sense all of these constitute 'rules of business,' but they have not covered a very important aspect ~ the set of rules that determine how a business operates ~ that is, rules that prevent, cause, or suggest things to happen.

For example, an entity/relationship diagram can represent the inherent operating structure of an organization. It can represent what is fundamentally possible in an organization. Depending on how it is constructed, it may also represent some of the constraints on an enterprise. As the model becomes more abstract, however, it describes fewer of these.

For example, as shown in Figure 1, a DIVISION may be *composed of* one or more SECTIONS, and a SECTION may be *composed of* one or more DEPARTMENTS. This is a very particular way of modeling an organization and expresses some very specific business rules. For example, a DEPARTMENT may not directly be *part of* a DIVISION, and a SECTION may not be *part of* a DEPARTMENT. This model has the disadvantage, however, that if these business rules change, the structure of the model must also change, as would the structure of any database which is based on that model. If the business were to define a new organization type (such as 'GROUP') between SECTION and DEPARTMENT, a new entity (and its corresponding table) would have to be added to the model, and the relationships would have to be re-wired.

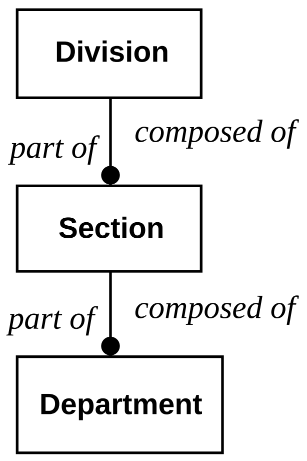


Figure 1: A Model of an Organization

An alternative model, shown in Figure 2, removes the explicit and arbitrary (that is, manmade) declarations of organization types, yielding a structure that can easily accommodate changes to the organization without itself having to be changed. In this version, an organization (*any* organization) may be part of any other organization. Furthermore, each organization may be identified as being of an organization type. Each organization type may also be part of another organization type. Now if a new organization type is added, it is simply another occurrence of organization type without any change to the structure. Thus, the possibilities for the enterprise have been opened up so that any system built based on the model will not itself impose constraints.

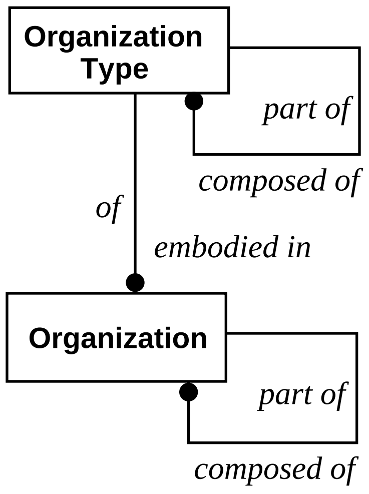


Figure 2: Another Model of an Organization

However, the business still does need to impose constraints, and those which were implicit in Figure 1 are no longer present in Figure 2. While such constraints should not be implemented via an inflexible database structure, they must still be accounted for in some fashion. A preferred method would be for us as analysts to describe such rules explicitly and individually. For example, we would like to document the rule that ‘cycles’ are not permitted[1] and the rule that the hierarchy of an organization must be consistent with the hierarchy of its types.[2]

Moving the expression of the constraints out of the model is not necessarily bad news. Previously, when these kinds of rules were implicit in our models, we tended not to be aware of them, making it harder to challenge them and ask if they were in fact appropriate. Now this becomes an explicit part of the analysis process.

In summary, constraints can be dynamic and changing, so it is not appropriate to build them routinely into implementation database structures. Even so, they are still important and must be accommodated. The Business Rules Project has set out to provide the means for making all business rules ~ not just the structural ones ~ explicit.

A CONTEXT FOR BUSINESS RULES

John Zachman has provided a useful context for discussing the architecture of an information system. His ‘Zachman Framework’ is a matrix which describes the various ways the stakeholders of an enterprise view the business and its systems.[3] It characterizes *architecture* in terms of the perspectives of the different stakeholders (represented by rows in the matrix) and focuses on the different aspects of architecture (represented by the columns). The rows represent, successively, the planner, owner, designer, builder, and subcontractor perspectives. The columns reflect the aspects of data, process, location, role, timing, and motivation.

The Business Rules Project has chosen initially to address the first and sixth aspects (i.e., the ‘data’ and ‘motivation’ columns) from the designer’s perspective (i.e., ‘row three’). In other words, the domain of the current effort is specifically the structure of a business’ data, along with the constraints and other motivation-related aspects of the enterprise information system. While documenting ‘data’ is reasonably well understood, the Business Rules Project is adding the aspect of ‘motivation.’ There will be some reference to the aspect of ‘process’ in business rules, but this will be minimal.

The Project’s position in ‘row three’ means that it is concerned with those business rules that affect the storage of persistent data values, described in a technology-neutral way. This stage of the Project is not concerned with the rules of a business that do not have an information system component.

DEFINITION OF A BUSINESS RULE

A business rule is a statement that defines or constrains some aspect of the business. It is intended to assert business structure or to control or influence the behavior of the

1 That is, ORGANIZATION A may not be *part of* ORGANIZATION B, if ORGANIZATION B is *part* of ORGANIZATION A, either directly or indirectly.

2 That is, if ORGANIZATION A is *part of* ORGANIZATION B, then the ORGANIZATION TYPE that ORGANIZATION A is *of* must be *part of* the ORGANIZATION TYPE that ORGANIZATION B is *of*; etc.

3 John A. Zachman, “A Framework for Information Systems Architecture,” *IBM Systems Journal*, Vol. 26, No. 3, 1987.

business. The business rules which concern the project are atomic ~ that is, they cannot be broken down further.

For our purposes, this may be viewed from two perspectives. From the business ('Zachman row two') perspective, it pertains to any of the constraints that apply to the behavior of people in the enterprise, from restrictions on smoking to procedures for filling out a purchase order. From the information system ('Zachman row three') perspective, it pertains to the facts which are recorded as data and constraints on changes to the values of those facts. That is, the concern is what data may or may not be recorded in the information system.

The GUIDE Business Rules Project decided to address the information system perspective first, and this paper reflects that orientation. Accordingly, a business rule expresses specific constraints on the creation, updating, and removal of persistent data in an information system. For example, the record of a purchase order may not be entered if the customer's credit rating is not adequate. While this may appear closely related to the business ('row two') rule which says that "we will not sell anything to a customer with a bad credit rating," the perspectives are subtly different.

Adopting this constrained scope provided three practical benefits to the project:

First, the focus on the information system perspective has made the problem tractable. It has been possible to understand and model business rules as constraints on data. It will be much more difficult to make equally clear assertions about business rules as they are practiced in the business.

Also, the project has intentionally not dealt with entire categories of issues pertaining to the behavior of people in an organization. These include:

- capturing the softer rules that involve the use of human judgment,
- looking at the other components of systems that are related to soft rules, and seeing how they come together to support the business rationale,
- showing work flows, processes, etc. in terms of their relationships to business rules,
- the process of acquiring, maintaining and enforcing rules,
- other systems of classification, such as:
  - mandates, policies, guidelines and corporate culture,
  - industry rules and corporate rules,
- determining when rules are ineffective.

It is the project team's intention that a follow-on project will address these issues, as well as delving deeper into the issues of inferences and action influencing assertions.

Finally, by dealing solely with issues of information structure, the project has not had to deal explicitly with events. From the perspective of an information system, an event only manifests itself by virtue of the fact that persistent data have come into existence or been modified. The business rules described here control whether or not such data may be created or changed, and the implications when this occurs.

CATEGORIES OF BUSINESS RULE

A statement of a business rule falls into one of four categories:

- ***Definitions of business terms***

  The most basic element of a business rule is the language used to express it. The very definition of a term is itself a business rule which describes how people think and talk about things. Thus, defining a term is establishing a category of business rule.

  Terms have traditionally been documented in glossaries or as entities in an entity/relationship model.

- ***Facts relating terms to each other***

  The nature or operating structure of an organization can be described in terms of the facts which relate terms to each other. To say that a customer can place an order is a business rule. Facts can be documented as natural language sentences or as relationships, attributes, and generalization structures in a graphical model.

  Facts and terms are discussed together in Chapter Four as 'Structural Assertions.'

- ***Constraints (here called 'action assertions')***

  Every enterprise constrains behavior in some way, and this is closely related to constraints on what data may or may not be updated. To prevent a record from being made is, in many cases, to prevent an action from taking place.

  Constraints are discussed in Chapter Five as 'Action Assertions.'

- ***Derivations***

  Business rules (including laws of nature) define how knowledge in one form may be transformed into other knowledge, possibly in a different form.

  Derivations are discussed in Chapter Six as 'Derivations.'

2
Formalizing Business Rules

Business rules can appear in many guises. They may be described in many different forms, both formal and informal. It is the purpose of the Business Rules Project to provide a basis for stating an organization's business rules formally and rigorously. With this in mind, certain underlying principles apply:

Explicit expression — The statement of business rules needs an explicit expression, either graphically or as a formal (logic-based) language. Currently, modeling notations are available to express some of the kinds of business rules. For example, structural rules may be represented by any of several flavors of entity/relationship (or 'object class') diagrams. Responses to events may be shown via essential data flow diagrams[4] or as entity life history diagrams.[5] There are fewer notations available, however, to describe action assertions. Most notable of these few are Object Role Modeling (ORM)[6] ~ derived from the Nijssen Information Analysis Method (NIAM) ~ and the Ross Method,[7] discussed later in this paper. Others could be developed.

Coherent representation — As the formalization of business rules becomes part of the commonly practiced systems analysis process, it is desirable for there to be a single, coherent representation for all the kinds of business rules. This would yield a single formal representation of all aspects of an enterprise's structure and operations. While this sounds overly ambitious, it should be reasonable, as a minimum, to seek more coherent links between the notations now in use for entities, event responses, and constraints, even if the individual notations are not changed.

Evolutionary extension — Conceptually, the representation of constraints can be thought of as an extension, for example, to entity/relationship diagram notation ~ adding constraints to the structural elements of a diagram. After all, the entities in an entity/relationship diagram represent things of significance to an organization, and it is reasonable for constraints to be described in terms of those things. Yet, while integration of concepts is a desirable goal, achieving an integrated *representation* was not the focus of this Business Rules Project. Our document is intended to describe the *nature* of business rules, regardless of how they might be portrayed.

---

4 Stephen M. McMenamin and John F. Palmer, *Essential Systems Analysis.* Englewood Cliffs, NJ:Yourdon Press, 1984.

5 Keith Robinson and Graham Berrisford, *Object-oriented SSADM*. Englewood Cliffs, NJ:Prentice Hall, 1994.

6 Terry Halpin, *Conceptual Schema & Relational Database Design, Second Edition*, Sydney:Prentice Hall Australia, 1995.

7 Ronald G. Ross, *The Business Rule Book: Classifying, Defining and Modeling Rules*, Boston, Massachusetts:Database Research Group, Inc., 1994.

Declarative nature **—** Note that a business rule is declarative, not procedural. It describes a desirable, possible state that is either suggested , required or prohibited. It may be conditional ~ that is, if something is the case, something else must or must not be the case. It does not, however, describe the steps to be taken to achieve the transition from one state to another, or the steps to be taken to prohibit a transition.

THE BUSINESS RULES CONCEPTUAL MODEL

To express the concepts and structure of a business rule formalism, the Business Rules Project has described the structure of a business rule itself as a conceptual model, in the form of an entity/relationship (E/R) diagram. That is, the model presented in this document defines what a business rule is and what kinds of rules there are.

The IDEF1X notation was chosen for the business rule model notation since it is a documented, public information modeling standard,[8] but other conceptual modeling conventions could also have been used. Appendix B provides an overview of the notation used in this paper.

The business rules model is organized around the following topics, presented in the next four chapters of this document:

- The origins of atomic business rules
- Structural assertions (terms and facts)
- Action assertions (constraints)
- Derivations

8 *Integration Definition for Information Modeling (IDEF1X).* FIPS PUB 184. National Institute of Standards and Technology (NIST): 1993.

3
Formulating Business Rules

The process of identifying business rules is often iterative and heuristic, where rules begin as general statements of policy. Even if the policy is formal and specific, it is typically described in a general and informal fashion, and it often remains for the employees to translate it into meaningful specific statements of what to do. Yet, even these more specific statements are still often in the nature of *business ramblings*, without discipline. Indeed, these statements may only sometimes originate in policy. More often, they arise from the day-to-day operation of the organization. As Barbara von Halle explains *rambling*, it implies "that these sentences are sometimes clear, sometimes (perhaps deliberately) ambiguous, and most of the time, contain more than one idea."[9] In spite of these shortcomings, *business ramblings* are usually an analyst's starting point for deriving more formal statements of business rules.

Initially, the analyst's assignment is to decompose these composite *ramblings* into *atomic* business rules, where each atomic business rule is a specific, formal statement of a single term, fact, derivation, or constraint on the business. Among other things, the analyst should evaluate the stability of the rule. Is it a fundamental aspect of the business, or is it likely to change in the near (or distant) future? Fundamental aspects may be seen in the company's infrastructure, while more transient business rules are often manifested in work practice.

Next, the task is to identify the atomic statement as the definition of a term, fact, constraint, or derivation. Terms, facts, and some of the constraints can be represented directly on graphical models. The remaining constraints and derivations must be translated into some other formalism. This can be as simple as natural language statements or it can have some more formal expression, such as a logic language specification or a graphical notation like that proposed by Ron Ross.[10] Whatever the form, ultimately the designer will be charged with identifying an appropriate technology for implementing the business rules in an information system.

9 Barbara von Halle, "Back to Business Rule Basics," *Database Programming & Design*, October 1994, pp. 15-18.

10 Ronald G. Ross, *op. cit.*

THE ORIGINS OF BUSINESS RULES — THE MODEL

Figure 3 shows the first part of the Business Rules conceptual model. (The full model may be found in Appendix A.) In the text above, terms like 'constraint,' 'rule,' and 'business rule' have been used in their conventional English sense. In the descriptions which follow, each of these terms and others are given very precise definitions, which are critical to the meaning of the model as a whole.

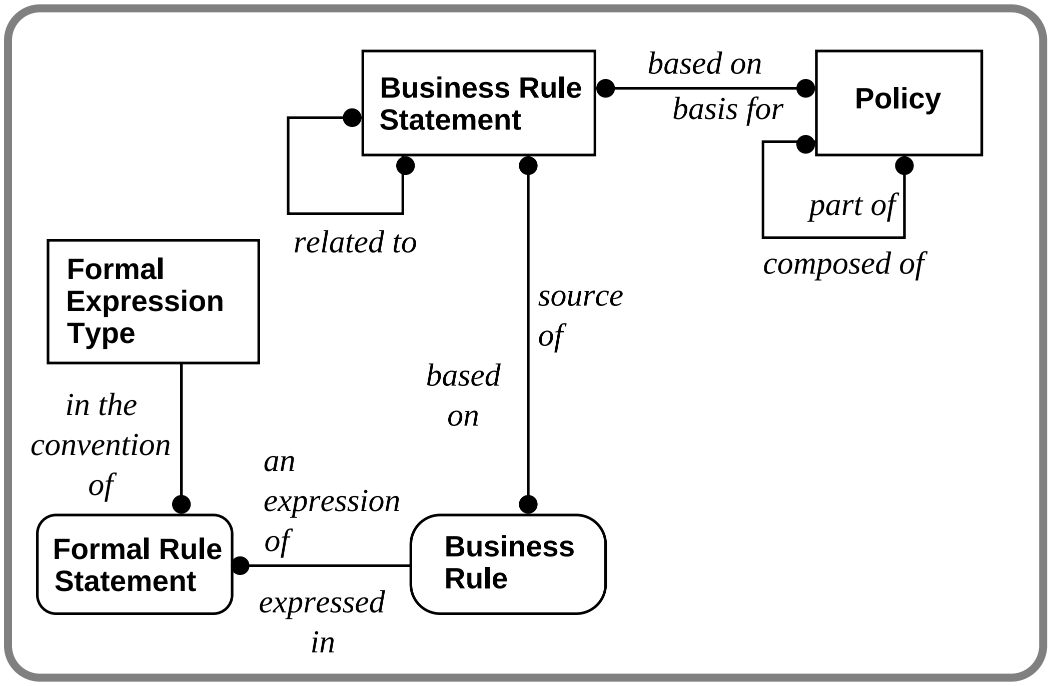


Figure 3: The Origin of Business Rules

In the model diagram, we see a POLICY — a general statement of direction for an enterprise. Each POLICY may be *composed of* more detailed POLICIES, which is to say, a detailed POLICY may be *part of* one or more general POLICIES.

An example of a POLICY for our car rental business might be:

- *”We only rent cars in legal, roadworthy condition to our customers.”*

A POLICY may be *the basis for* one or more BUSINESS RULE STATEMENTS (‘business ramblings’), just as a BUSINESS RULE STATEMENT may be *based on* one or more POLICIES. A BUSINESS RULE STATEMENT is a declarative statement of structure or constraint which the business places upon itself or has placed upon it. Each BUSINESS RULE STATEMENT may be *related to* one or more other BUSINESS RULE STATEMENTS.

For example, each of the following could be a BUSINESS RULE STATEMENT :

- *“Cars should be checked on return from each rental, and on transfer between branches.”*
- *“If any lights are not working, the bulbs should be replaced. If tires are worn, they should be replaced.”*
- *“Under any of the following conditions the car should be scheduled for service or repair:*
  - — *accumulated mileage since the last service is greater than 5000,*
  - — *the brakes are not satisfactory,*
  - — *the exhaust is noisy or emitting fumes,*
  - — *there is any damage to body work (apart from superficial dents and scratches), lights or glass,*
  - — *there are any significant fluid leaks.”*

A BUSINESS RULE STATEMENT, in turn, may be the *source of* one or more (ATOMIC) BUSINESS RULES. Like a BUSINESS RULE STATEMENT, a BUSINESS RULE is a statement that defines or constrains some aspect of the business, but (in contrast to a BUSINESS RULE STATEMENT) it cannot be broken down or decomposed further into more detailed business rules. If reduced any further, there would be loss of important information about the business. Each BUSINESS RULE may be *based on* one or more BUSINESS RULE STATEMENTS.

For example, a BUSINESS RULE might be:

- *"A car with accumulated mileage greater than 5000 since its last service must be scheduled for service."*

It is important to note that an enterprise's business rule applies, regardless of the form used to express it. Business rules have been in place and companies have been responding to them long before anyone ever dreamed of formalizing them or drawing pictures of them. Business rules are an underlying reality in an organization — independent of an analyst's attempt to structure and describe them.

Note also that each BUSINESS RULE may be *expressed in* one or more FORMAL RULE STATEMENTS, although each FORMAL RULE STATEMENT must be an *expression of* just one (ATOMIC) BUSINESS RULE. A FORMAL RULE STATEMENT is an expression of a BUSINESS RULE in a specific formal grammar. A FORMAL RULE STATEMENT must be *in the convention of* a particular FORMAL EXPRESSION TYPE, which is to say one of the formal grammars for representing BUSINESS RULES. Examples of a FORMAL EXPRESSION TYPE are structured English, IDEF1X, Oracle's CASE*Method, Object Role Modeling, Ross's notation, and so forth.

A structured English example of a FORMAL RULE STATEMENT is:

-
  ```
  If Car.miles-current-period > 5000 then
      invoke Schedule-service (Car.id)
  End if
  ```

TYPES OF BUSINESS RULE

Each BUSINESS RULE must be one of the following:

- A STRUCTURAL ASSERTION — a defined concept or a statement of a fact that expresses some aspect of the structure of an enterprise. This encompasses both terms and the facts assembled from these terms.
- An ACTION ASSERTION — a statement of a constraint or condition that limits or controls the actions of the enterprise.
- A DERIVATION — a statement of knowledge that is derived from other knowledge in the business.

Figure 4 shows the part of the business rule model reflecting these ideas. Each of these is described further in subsequent chapters: chapter 4 (STRUCTURAL ASSERTION), chapter 5 (ACTION ASSERTION), and chapter 6 (DERIVATION).

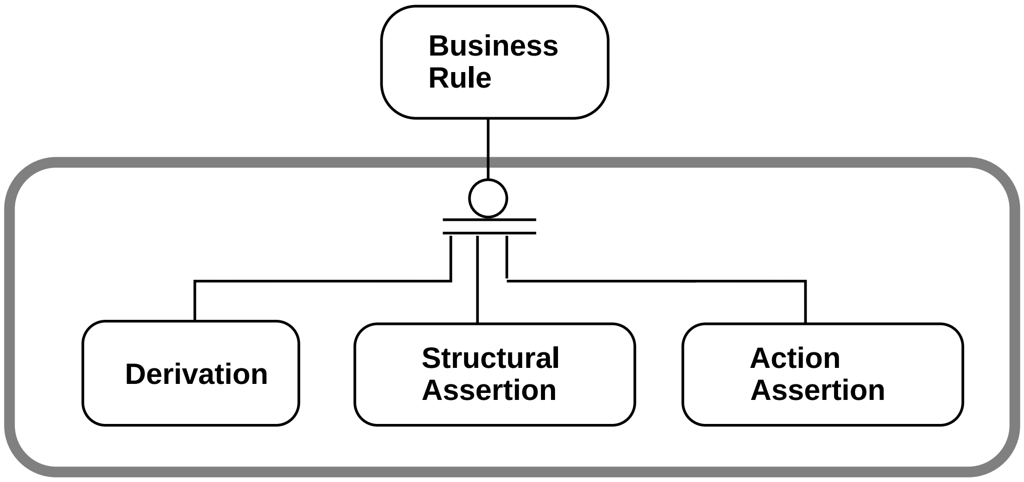


Figure 4: Business Rule Types

DEFINITIONS

The following summarizes the definitions of the concepts relating to *Business Rule*.

Definitions	
Business Rule	a statement that defines or constrains some aspect of the business. This must be either a term or fact (described below as a STRUCTURAL ASSERTION), a constraint (described below as an ACTION ASSERTION), or a DERIVATION. It is 'atomic' in that it cannot be broken down or decomposed further into more detailed business rules. If reduced any further, there would be loss of important information about the business.
Business Rule Statement	a declarative statement of structure or constraint which the business places upon itself or has placed upon it.
Formal Expression Type	one of the formal grammars for representing BUSINESS RULES.
Formal Rule Statement	an expression of a BUSINESS RULE in a specific formal grammar.
Policy	a general statement of direction for an enterprise.

Table 1: Business Rule Definitions

4
Structural Assertions

The first kind of business rule to be discussed is the STRUCTURAL ASSERTION. A STRUCTURAL ASSERTION is a statement that something of importance to the business either exists as a concept of interest or exists in relationship to another thing of interest. It details a specific, static aspect of the business, expressing things known or how known things fit together. STRUCTURAL ASSERTIONS are frequently portrayed by entity/relationship models.

TERMS AND FACTS

As shown in Figure 5, STRUCTURAL ASSERTIONS are of two kinds: TERMS and FACTS.

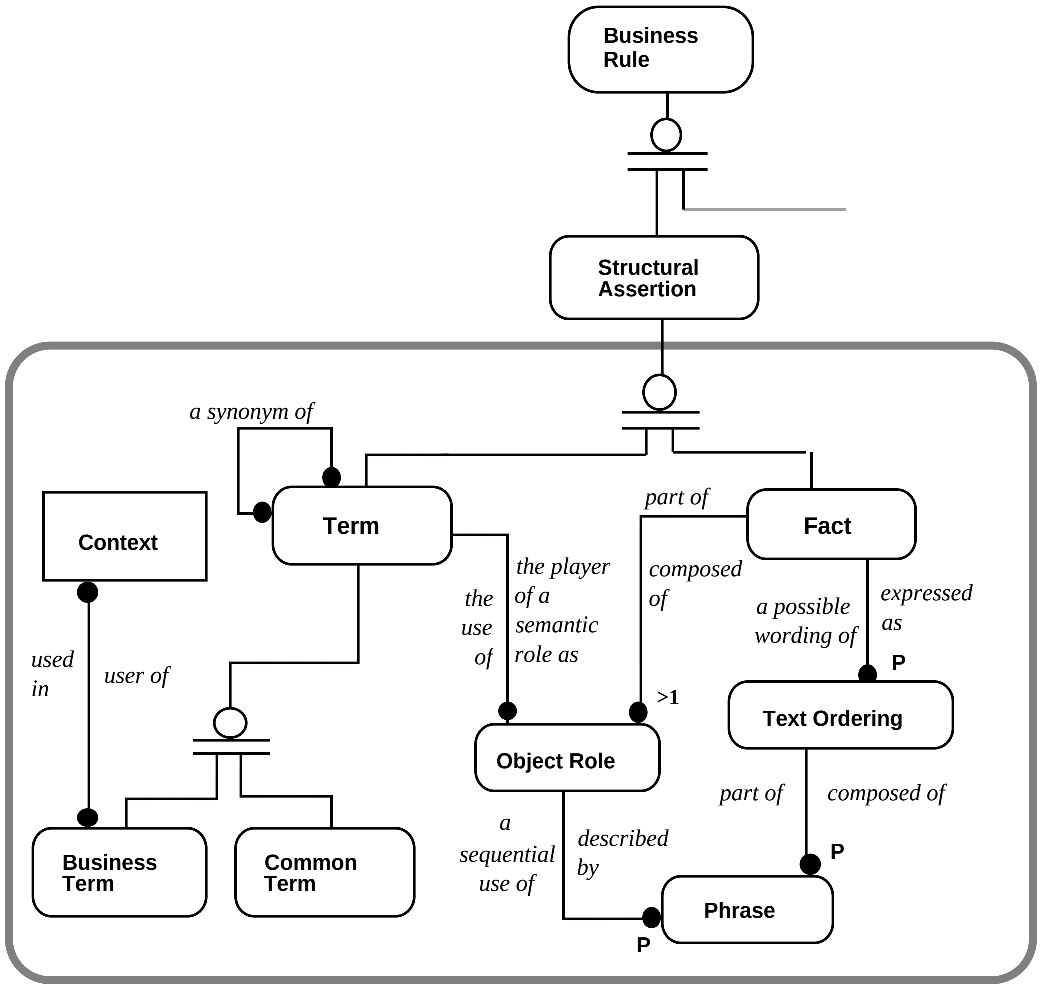


Figure 5: Terms and Facts

A TERM is a word or phrase which has a specific meaning for the business. The TERMS of interest here are of two types: BUSINESS TERMS and COMMON TERMS. A BUSINESS TERM is a word or phrase that has a specific meaning for a business in some designated CONTEXT. Each BUSINESS TERM must be *used in* at least one CONTEXT, and each

CONTEXT may be *the use of* one or more BUSINESS TERMS with some defined meaning attached.

For example, BUSINESS TERMS in the CONTEXT of EU-Rent's car rental business might include 'rental request,' 'reservation,' 'booking,' etc. COMMON TERMS are words in everyday language using their commonly-accepted meaning. Specifically, COMMON TERMS are part of the basic vocabulary, for example, 'car,' 'city,' etc., and are taken as axiomatic to avoid writing circular definitions. Both BUSINESS TERMS and COMMON TERMS are used as TERMS in constructing sentences, that is, statements of business FACTS.

The difference between a COMMON TERM and a BUSINESS TERM is that a BUSINESS TERM must be defined explicitly in terms of one or more FACTS. (Each FACT may be used in the definition of one or more BUSINESS TERMS.) The definition of a COMMON TERM is generally understood and need not be defined explicitly.

A FACT asserts an association between two or more TERMS. That is, it expresses a relationship between the TERMS. A FACT involves two or more TERMS, and a TERM may be in one or more FACTS. Note that a FACT is not limited to a simple binary pairing of TERMS. Indeed, some FACTS must be expressed by compound associations with more than two components. For example, "a customer may request a model of car from a rental branch on a date" is a FACT which includes four TERMS: customer, car model, rental branch, date. Also note that, in this case, the request is not for a particular car but for any car of a particular model. This FACT defines the BUSINESS TERM 'model rental request.'

An occurrence of OBJECT ROLE is needed to depict each semantic role that a TERM plays in a FACT. That is, each TERM may be *the player of a semantic role as* one or more OBJECT ROLES, and each OBJECT ROLE must be *the use of* one TERM *in* one FACT. For example, in a company's sales area the FACT that there is a relationship between the TERMS 'customer' and 'contract' could be established (shown graphically in Figure 6). There would be two OBJECT ROLES in this FACT — one asserting that the TERM 'contract' may be *the player of a semantic role as* an OBJECT ROLE that is *part of* the FACT and another asserting that the TERM 'customer' may be *the player of a semantic role as* an OBJECT ROLE that is *part of* the FACT.

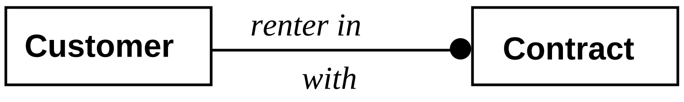


Figure 6: A Sample Fact

Note that while not all TERMS need to be reflected in some FACT, each *BUSINESS* TERM recorded ***must be*** *used in* one or more FACTS (i.e., as an OBJECT ROLE). Each FACT *must be composed of* two or more OBJECT ROLES.

A FACT often has multiple ways of being stated. Even in a binary FACT there can be at least two expressions of the FACT — one describing it in each direction. In the customer/contract example, it is both the case that "each CONTRACT may be *with* a CUSTOMER," and that "each CUSTOMER may be the *renter in* many CONTRACTS." These two sentences describe the same underlying FACT.

In the business rules model, this is represented by the entity TEXT ORDERING. That is, each FACT may be *expressed as* one or more TEXT ORDERINGS, where a TEXT ORDERING portrays *one possible wording of* a FACT. Each of the two sentences about contract/customer and each of the three sentences about model rental request would be a TEXT ORDERING about their respective FACTS.

In our car model example, the original sentence above could be restated in various other TEXT ORDERINGS, for example:

- *"A model may be requested by a customer from a rental branch on a date."*

  (Note that this wording is ambiguous ~ the date is the date the car is needed, not the date the request is submitted.)
- *"A rental branch may be requested to provide, on a given date, a car of a particular model to a customer."*

Each TEXT ORDERING must be *composed of* one or more PHRASES, where each PHRASE must be *a sequential use of* one of the OBJECT ROLES that is part of the FACT. (Conversely, each OBJECT ROLE of the FACT must be *described by* one or more PHRASES.) In addition to identifying the sequence of the OBJECT ROLE in the TEXT ORDERING, the PHRASE also provides the text (with a marker) and specifies the syntactic role of the OBJECT ROLE (e.g., subject, object).

For example, "Each CUSTOMER may be the *renter in* many CONTRACTS" is one TEXT ORDERING of the underlying FACT portrayed in Figure 6. The PHRASE 'each CUSTOMER' describes the OBJECT ROLE ('customer' doing the buying) as the 'subject' in this TEXT ORDERING of the FACT. This PHRASE further specifies that 'customer' occurs in 'position 1' in this ordering, and it provides the text phrase as "each <>" (where <> represents the marker for the TERM).

Then, in the second TEXT ORDERING for the same FACT ("Each CONTRACT may be *with* many CUSTOMERS"), the same TERM, 'customer,' appears in a different OBJECT ROLE ('customer' as the 'object' of the CONTRACT) and this OBJECT ROLE is *described by* another PHRASE — one which specifies it as the 'object' in the sentence, its position as '2,' and its text with marker as "may be with many <>."

Note the parallel structures between a FACT being *expressed as* one or more PHRASES (via TEXT ORDERINGS), and a FACT being *composed of* one or more OBJECT ROLES. While it is not shown graphically on the model, it is the case that there must be exactly one PHRASE which is *part of* a TEXT ORDERING for every OBJECT ROLE which is *part of* the FACT *expressed as* that TEXT ORDERING.

Note also that one attribute of TEXT ORDERING could be 'source.' That is, such an attribute would indicate whether an occurrence of TEXT ORDERING came from a user or not. TEXT ORDERINGS from users could readily be used to feed back the analysis to users: "This is what you said. Is it correct? Here is another way of saying the same thing. Do you agree?"

KINDS OF TERM

Previously, we classified a TERM into a BUSINESS TERM or a COMMON TERM. Figure 7 shows that a TERM may also be classified in another way — as a TYPE (defining abstract categories of things like 'car model,' 'walk-in rental,' 'customer,' etc.) or as a LITERAL (describing instances of things, such as 'General Motors,' or '5000'). A TYPE is a named abstraction of a set of instances or values while a LITERAL is a specific value or instance. Business rules most often are stated in terms of TYPES but occasionally need to make reference to specific values or instances.

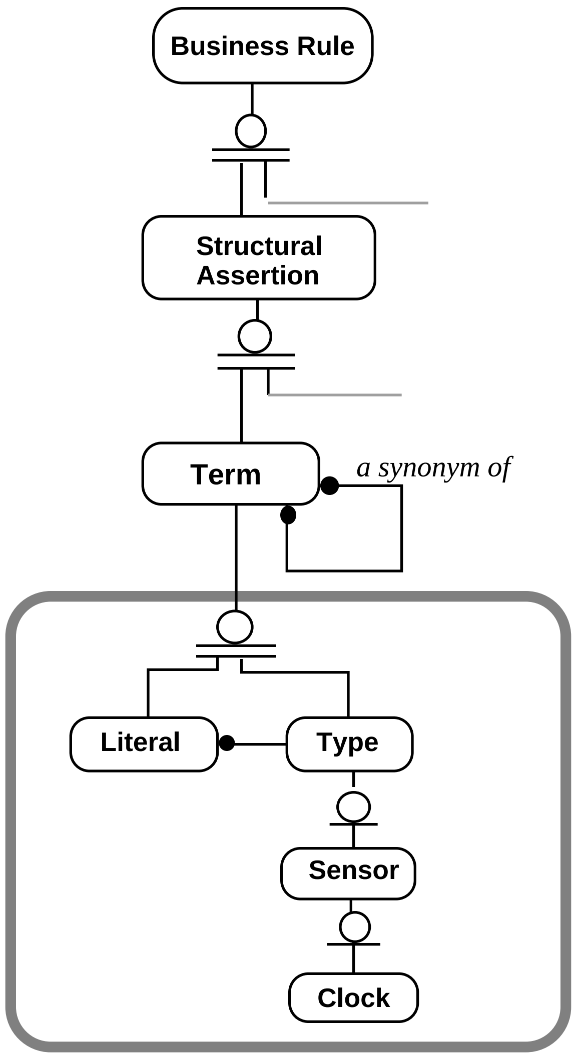


Figure 7: Kinds of Term

A particular kind of TYPE is SENSOR. A SENSOR represents the presence of something that detects and reports constantly changing values from the outside world, such as a temperature reading, or some other value. In our rental car example, a SENSOR could be the device at a parking lot which records the time when a car enters or leaves the lot. A special type of SENSOR is a CLOCK, which reports the passage of time. It always has one value, the 'current time.'

Finally, we will postulate that for any given concept, a single TERM is designated as the word which labels it. (This assumes, for this discussion, a single natural language such as English). Given this 'base' TERM as a reference point, we can then assert that other TERMS may be synonyms of this base term. That is, each TERM may have one or more other TERMS as its synonyms. Conversely, each TERM may itself be *a synonym of* one or more other TERMS.

KINDS OF FACT

FACTS may be classified in two different ways, as shown in Figure 8. The first classification distinguishes between a BASE FACT (which is simply asserted) and a DERIVED FACT (whose value is computed or inferred from other BUSINESS RULES). The second classification distinguishes a FACT as one of three types: ATTRIBUTE, PARTICIPATION, or GENERALIZATION.

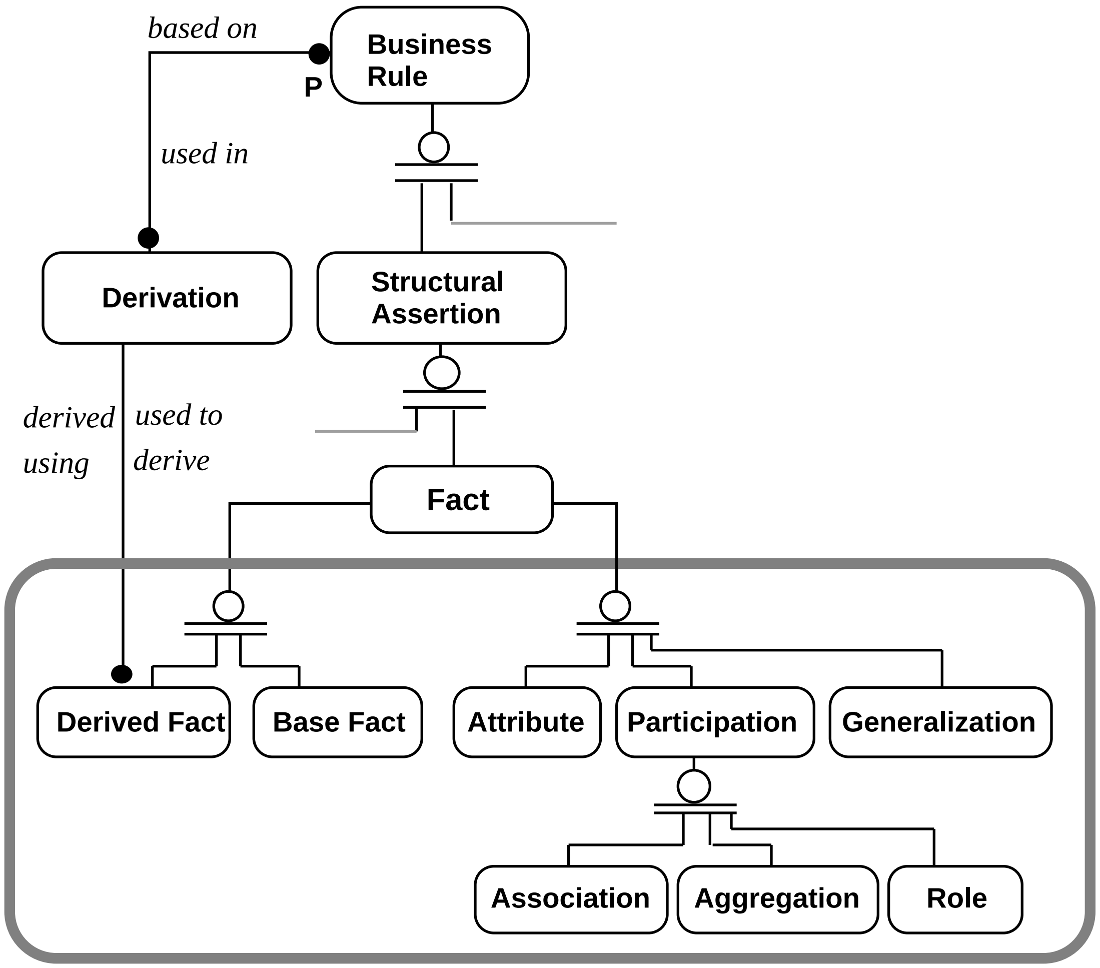


Figure 8: Kinds of Fact

Base / Derived

A FACT may be either a BASE FACT or a DERIVED FACT. A BASE FACT is simply recorded as something that is the case. A DERIVED FACT is an assertion that is constructed from other assertions. It may be a computed value, or it may be a 'view.' For example, one fact may be that 476 is recorded at point 321-C in a processing plant at 3:46 p.m. on January 14. From this can be derived the fact that 476 gallons of kerosene are in inventory in tank 234 at 3:46 p.m. on January 14. Similarly, measurements may be taken for volume and flow past a processing point, and from this is derived the mass of material that passed that point during a specified period of time.

Each DERIVED FACT must be *derived using* a DERIVATION — an inference or a mathematical calculation that may be either a MATHEMATICAL CALCULATION (as in the 'mass' example) or an INFERENCE (as in the inventory example). Each DERIVATION must be *based on* one or more other BUSINESS RULES, and a BUSINESS RULE may be *used in* any number of DERIVATIONS. Depending on the approach taken, properties of DERIVATION could be the formula involved or simply the name of an algorithm which carries out the derivation.

In the car rental example, a BASE FACT could be:

- *"A car model (e.g., Mercury Mystique) is in a car class (e.g., Class C)."*

DERIVED FACTS could be:

- *"Rental charge is based on base rental price, optional insurances, and refueling charge."*
- *"The number of cars (of a group) that will be available the next day to meet demand is computed as the number of cars of that group currently in the parking lot, plus the number due in today from rental." For example, there are 4 group B cars in the parking lot, and 7 are due from rental today, so there should be 11 available to meet demand for tomorrow."*
- *"Base rental price for a car is the rate for the group that car's model belongs to."*
- *"Number of rentals, turnover and profit of a branch in the past year can determine the targets for that branch for the next quarter."*

A DERIVATION *used to derive* the first of these DERIVED FACT would be:

- *Rental charge = Base rental price + Optional insurances + Refueling charge*

DERIVED FACTS and DERIVATIONS are discussed further in Chapter 6.

Attribute / Participation / Generalization

The second classification distinguishes a FACT as one of three types:

- the designation of a TERM as an ATTRIBUTE of another TERM,
- the designation of a TERM as a GENERALIZATION (or, supertype) of one or more other TERMS (its subtypes), and
- the designation of a TERM as a PARTICIPATION (or relationship) between other TERMS.

An ATTRIBUTE expresses a FACT in which a TERM describes some aspect of another TERM. A GENERALIZATION expresses a type of FACT in which one TERM (a TYPE) describes a subset of occurrences of another TERM (also a TYPE).

Example ATTRIBUTES for our rental car business might be:

- *"Name is an attribute of customer."*
- *"Color is an attribute of car."*

Example GENERALIZATIONS might be:

- *"A rental branch manager is an employee."*
- *"A branch is a EU-Rent location."*

A PARTICIPATION expresses a fact in which a set of terms are associated in some sense which is meaningful to the business ~ for example, "a car model may be requested by many customers," or "a car may be rented by many customers," (but only one at a time). This is a relationship of the sort that would typically appear in an entity/relationship model, probably further augmented by a constraint, such as 'one or more' or 'no more than one.'

A PARTICIPATION, in turn, is one of three general kinds:

- An AGGREGATION is a ‘part of/composed of’ relationship.
- A ROLE describes how one TERM serves as an actor (another TERM) through its interactions with its environment.
- An ASSOCIATION is simply any other kind of relationship.

Example PARTICIPATIONS (AGGREGATIONS) for our rental car business might be:

- *“A rental group is composed of car models.”*
- *“The branch inventory of a car model is composed of the cars of that model owned by the branch.”*

Example PARTICIPATIONS (ROLES) might be:

- *“A person may be the additional driver in a rental.”*
- *“A branch may be the car gainer in a transfer.”*
- *“A branch may be the car loser in a transfer.”*

Example PARTICIPATIONS (ASSOCIATIONS) might be:

- *“Car models may be requested by customers.”*
- *“Cars may be rented by customers.”*

DEFINITIONS

The following summarizes the definitions of the concepts relating to *Structural Assertion*.

Definitions	
Aggregation	a specialization of PARTICIPATION that expresses an 'is part of / is composed of' FACT. The TERMS in the FACT describe the component types of the whole.
Association	a specialization of PARTICIPATION that expresses an 'is associated with' FACT between two or more TYPEs. Ideally, an ASSOCIATION is described using a verb phrase that expresses the particular nature of the association. It represents any type of PARTICIPATION (relationship) other than AGGREGATION or ROLE.
Attribute	a specialization of FACT that expresses a 'has property of' relationship between TERMs, specifically an association between an entity type and a domain/abstract data type. For example, "a customer has a name."
Base Fact	a FACT that is not a DERIVED FACT.
Business Term	a word or phrase that has a specific meaning for a business in some designated CONTEXT.
Clock	a special kind of SENSOR whose value is 'real-world time.' A CLOCK must have exactly one value, which is 'current time.'
Common Term	a word or phrase in everyday language using its commonly-accepted meaning. Specifically, common terms are part of the basic vocabulary, for example, 'year,' 'calendar,' etc., and are taken as axiomatic to avoid writing circular definitions.
Context	an environment where shared BUSINESS TERMs are used with an agreed-to meaning.
Derived Fact	a FACT whose value is computed or inferred from other FACTS via a specified DERIVATION.
Fact	an associating of two or more TERMS. It expresses a <u>potential</u> for association ('can be' or 'may be') rather than expressing a 'must be' association.
Generalization	a specialization of FACT in which one TERM (a TYPE) describes a subset of occurrences of another TERM (also a TYPE).
Literal	a kind of TERM that reflects a specific value or instance of a TYPE.
Object Role	the semantic role that a TERM plays in a FACT.
Participation	any association between TERMS other than ATTRIBUTE or GENERALIZATION. (This would typically appear as a relationship in an entity/relationship diagram.)
Phrase	one component of a TEXT ORDERING that reflects the usage of an OBJECT ROLE in a specific position of the TEXT ORDERING, identifying its syntactic role and providing its portion of the text (with marker).
Role	a statement of the way in which one TERM may serve as an actor (another TERM) through its interactions with its environment. For example, "a customer may be *a* buyer *in* a contract." *(or, a* seller *in, the* recipient *of,* etc.)

Sensor	a special kind of TYPE whose value is asserted by some mechanism or device whose inner workings are unknown (or uninteresting to the identified scope). Its value cannot be altered directly. A SENSOR detects and reports constantly changing values from the outside world, such as the passage of time, a temperature reading, or some other value.
Structural Assertion	a statement that something of importance to the business either exists as a concept of interest or exists in relationship to another thing of interest. It details a specific, static aspect of the business, expressing things to be known or how known things fit together.
Term	a word or phrase used by the business.
Text Ordering	the portrayal of one possible wording of a FACT. It is an aggregation of its constituent PHRASES.
Type	a kind of TERM that is a named abstraction of a set of instances or values.

Table 2: Structural Assertion Definitions

5
Action Assertions

An ACTION ASSERTION is a statement that concerns some dynamic aspect of the business. It specifies constraints on the results that actions can produce. The constraints are described non-procedurally, in terms of the other atomic business rules. Where the STRUCTURAL ASSERTIONS describe possibilities, ACTION ASSERTIONS impose constraints — 'must' (or, 'should') or 'must not' (or, 'should not'). Figure 9 shows the business rule model of ACTION ASSERTIONS.

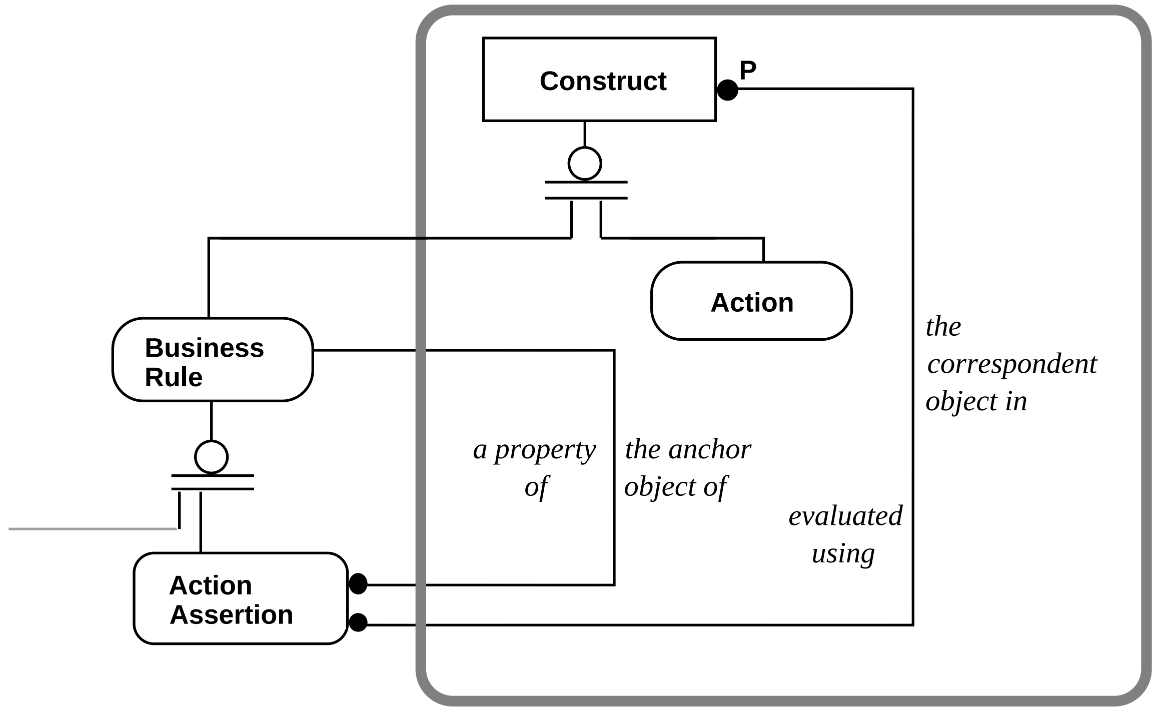


Figure 9: Action Assertions

ACTION ASSERTION COMPONENTS

An ACTION ASSERTION is composed of an anchor object and one or more correspondent objects, possibly with one or more modifiers.[11] The *anchor* ***object*** may be any kind of BUSINESS RULE. Specifically, each ACTION ASSERTION must be *a property of* one other BUSINESS RULE. In turn, that BUSINESS RULE may be *the anchor object of* one or more ACTION ASSERTIONs. In the phrase, "If . . . then . . ." this is the phrase after 'If.' The *anchor object* is often a STRUCTURAL ASSERTION, but it may also be another ACTION ASSERTION.

A *correspondent* ***object*** may be either another BUSINESS RULE or some specified ACTION, represented by the generalized term CONSTRUCT, where a CONSTRUCT is either some

11 Much of the material presented in this chapter was derived from Ronald G. Ross's book, previously cited.

other BUSINESS RULE or an ACTION. That is, a CONSTRUCT is *the correspondent object in* an ACTION ASSERTION. An ACTION could be the sale of a car, rental of a car, or suspension of a customer. Each of these things could be *the correspondent object in* an ACTION ASSERTION. An ACTION ASSERTION is *evaluated using* one or more CONSTRUCTS. These describe the conditions doing the constraining. In the phrase, "If . . . then . . ." each of these would go after 'then.'

For example, ACTION ASSERTIONS might be:

- *"A car must have a registration number."*

  The BUSINESS RULE 'car' (which is a TERM) is the *anchor object* of this ACTION ASSERTION.

  The BUSINESS RULE that expresses the possibility that a car may have a registration number (a FACT) is the *correspondent object* of this ACTION ASSERTION.

- *"A car cannot be handed over to the customer unless a provisional charge has been accepted against* the customer's *credit card."*

  The handing over of a car is recorded as a state change on a rental. This resultant state is a TERM that is the *anchor object* of the ACTION ASSERTION.

  The verification that a provisional charge has been accepted against the customer's credit card is the BUSINESS RULE (a FACT) that is the *correspondent object* of the ACTION ASSERTION.

ACTION ASSERTION CLASSIFICATIONS

ACTION ASSERTIONS may be classified in three ways. First, an ACTION ASSERTION ***class*** identifies whether an ACTION ASSERTION is either a CONDITION, an INTEGRITY CONSTRAINT, or an AUTHORIZATION. An ACTION ASSERTION may also be classified into one of a larger number of ACTION ASSERTION ***types***, which define the specific nature of the constraint. Finally, an ACTION ASSERTION may be defined as an ACTION CONTROLLING ASSERTION or an ACTION INFLUENCING ASSERTION. The remainder of this chapter explains and illustrates these three classifications of ACTION ASSERTION.

Note that the first action assertions that will be apparent are those which can be depicted graphically on an entity/relationship model, such as those that constrain a relationship to applying to 'at least one' or 'no more than one' occurrence of an entity. Similarly, attributes may be constrained as 'mandatory.' Other model graphics permit subtypes to be specified as 'mutually exclusive' and subtype clusters to be designated as 'complete.'

ACTION ASSERTION CLASSES

As shown in Figure 10, every ACTION ASSERTION must be classified as either an AUTHORIZATION, a CONDITION, or an INTEGRITY CONSTRAINT.

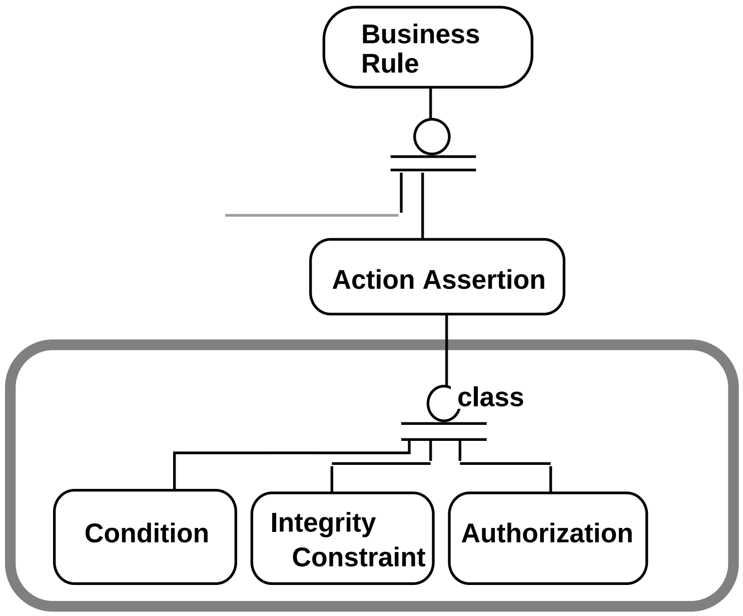


Figure 10: Classes of Action Assertion

A CONDITION is an assertion that *if* something is true, another business rule will apply. It can be thought of as a 'test' ~ if true, it may be the basis for enforcing or testing other ACTION ASSERTIONS. For example, a CONDITION can ask: "did a customer not show a valid driver's license?" "is a customer in arrears?" or "has a customer placed an order?"

An INTEGRITY CONSTRAINT is an assertion that must always be true. It is considered to have immediate enforcement power because it prohibits any actions which would result in a false truth value. While a CONDITION can test for a value (e.g., ask "is a car registered?") and then specify some action based on that test, an INTEGRITY CONSTRAINT can declare that 'a car must be registered' and prohibit any action which would result in violation of that end state. Such an integrity constraint, for example, would prohibit both creating a new car instance without a registration value, as well as setting an existing car's registration to 'null.'

An AUTHORIZATION defines a specific prerogative or privilege with respect to one or more CONSTRUCTS. It is an assertion represented by the predicate:

(Only) *x* may do *y*,

where *x* typically is a user and *y* is an action that may be executed or performed. AUTHORIZATIONS are given only to TYPES capable of independent activity (e.g., people, departments, computers, etc.). For example, only a branch manager of the 'losing' branch may assign a car for transfer to another branch.

ACTION ASSERTION TYPES

As shown in Figure 11, every ACTION ASSERTION may also be classified by type. The business rules model depicts only three possible types — ENABLER, TIMER, and EXECUTIVE. This list is by no means exhaustive. Appendix C presents one view of a more comprehensive set of types than that discussed by the Project team or this paper.

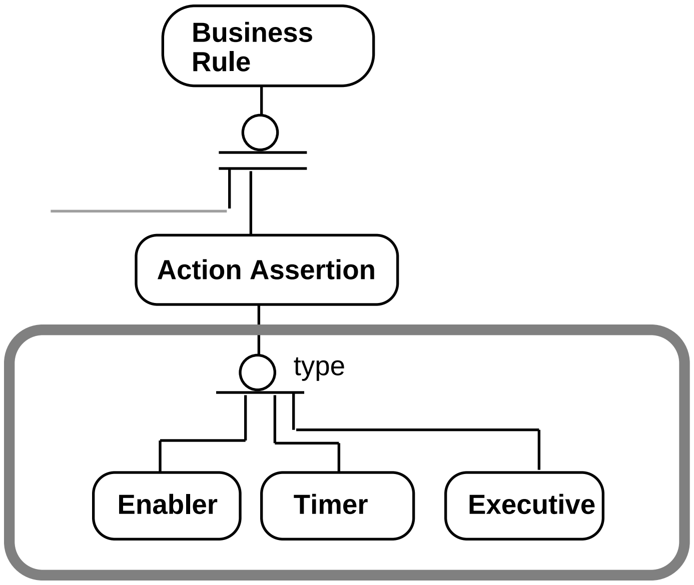


Figure 11: Types of Action Assertion

An ENABLER is a type of ACTION ASSERTION which, if true, permits or leads to the existence of the correspondent object. The assertion is true if the anchor object exists. This has varying interpretations depending on the nature of the correspondent object:

- If the correspondent object is a STRUCTURAL ASSERTION, this type of ACTION ASSERTION permits (i.e., enables) the creation of a new instance.
- If the correspondent object is another ACTION ASSERTION, this type of ACTION ASSERTION permits the other ACTION ASSERTION.
- If the correspondent object is an ACTION, this type of ACTION ASSERTION permits its execution (otherwise, its execution is blocked/disabled).

More precisely, the above description is of an ENABLER which is an INTEGRITY CONSTRAINT. When an ENABLER is a CONDITION, it simply tests to see if the correspondent object is enabled rather than directly enabling the correspondent object.

A TIMER is a type of ACTION ASSERTION which tests, enables (or disables), or creates (or deletes) if a specified threshold has been satisfied. A TIMER can be thought of as a 'countdown timer' ~ its effect occurs after the 'ticking' stops ~ or as an 'alarm clock.' In the latter case, its effect occurs when the alarm clock 'rings.' For example, if the thing that is controlled by a TIMER is another ACTION ASSERTION, then the TIMER will 'turn on' the ACTION ASSERTION. If the thing that is controlled by a TIMER is a STRUCTURAL ASSERTION (e.g., an ATTRIBUTE), the TIMER will set (or test) its value.

An EXECUTIVE is a type of ACTION ASSERTION which requires (causes) the execution of one or more ACTIONS. The following example shows how these types can be combined in various ways. In the statement "if a customer is three months in arrears, then repossess the car," the part that measures (counts down) 'three months in arrears' and requires action thereafter is a CONDITION of type TIMER. A second ACTION ASSERTION 'repossess the car' is an INTEGRITY CONSTRAINT of type EXECUTIVE.

CONTROLLING VS. INFLUENCING

All of the ACTION ASSERTION classes and types described above are intended to be used to define what *must (or must not) be* or what *must (or must not) happen.* When used in this way, they are examples of ACTION ***CONTROLLING*** ASSERTIONS. ACTION ASSERTIONS can also be used, however, to describe what should (or should not) be or what should (or should not) happen. These ACTION ***INFLUENCING*** ASSERTIONS tend to be things that companies are not inclined to build into computer systems as hard constraints. However, they may be of sufficient interest that management wants to be notified by the information system if they are violated. Or, they may simply serve as guidelines in the human activity system, with or without direct automation support.

Figure 12 shows ACTION CONTROLLING ASSERTIONS and ACTION INFLUENCING ASSERTIONS.

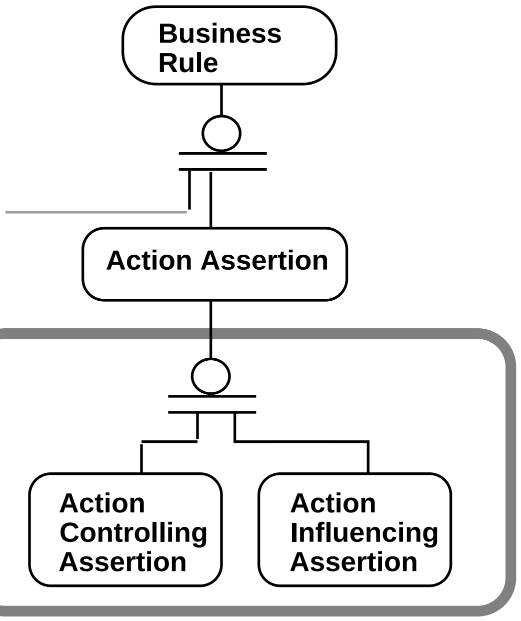


Figure 12: Action Controlling vs. Action Influencing Assertions

The idea of ACTION INFLUENCING ASSERTIONS argues for a particular orientation for decision support information. Companies may need to specify rules with their decision support information, to provide guidance:

- to determine that action influencing constraints have been violated.

  For example, "in each quarter, no more than 10% of rentals should have free upgrades."

- to decide what to do when the rule is broken.

  For example, live with a one-off anomaly, change the mix of cars at the branch, or discipline the manager.

Meeting these requirements may require the definition of additional TERMS and FACTS.

Aside from identifying that there is an 'influencing' form of an ACTION ASSERTION, the Business Rules Project has not modeled these in detail in this stage of the Project.

DEFINITIONS

The following summarizes the definitions of the concepts relating to *Action Assertion*.

Definitions	
Action	something that executes and may change the state of one or more instances of one or more TYPEs. An ACTION has a protocol and one or more methods that implement it. An ACTION cannot be constrained; only TYPEs (things which have persistent instances) can be constrained. The enabling and execution of an ACTION can be controlled through rules . The ACTION is permitted to proceed once/if conditions are satisfied.
Action Assertion	a statement that concerns some dynamic aspect of the business. It specifies constraints on the results that actions can produce.
Action Controlling Assertion	an ACTION ASSERTION that describes what must or must not be (or happen).
Action Influencing Assertion	an ACTION ASSERTION that describes what should or should not be (or happen).
Authorization	an assertion that a specific prerogative or privilege has been defined with respect to one or more CONSTRUCTS. An AUTHORIZATION is an assertion represented the predicate: "(Only) *x* may do *y*," where *x* typically is a user and *y* is an action that may be executed.
Condition	an assertion that *if* something is true, another business rule will apply. A CONDITION can be thought of as a 'test' ~ if true, it may be the basis for enforcing or testing other ACTION ASSERTIONs.
Construct	a generalization that represents either a BUSINESS RULE or an ACTION.
Integrity Constraint	an assertion that must always be true.

Table 3: Action Assertion Definitions

6
Derivations

A BASE FACT is a FACT that is a given in the world and is remembered (stored) in the system. A DERIVED FACT is created by an inference or a mathematical calculation from TERMS, FACTS, other DERIVATIONS, or even ACTION ASSERTIONS.

Figure 13 shows the business rules model of DERIVATIONS. In this view, a DERIVED FACT is shown as a kind of FACT, along with BASE FACT. Each DERIVED FACT must be *derived using* one DERIVATION. The DERIVATION, in turn, must be *based on* one or more BUSINESS RULES. In other words, a DERIVATION also must be *used to derive* at least one and possible more DERIVED FACTS.

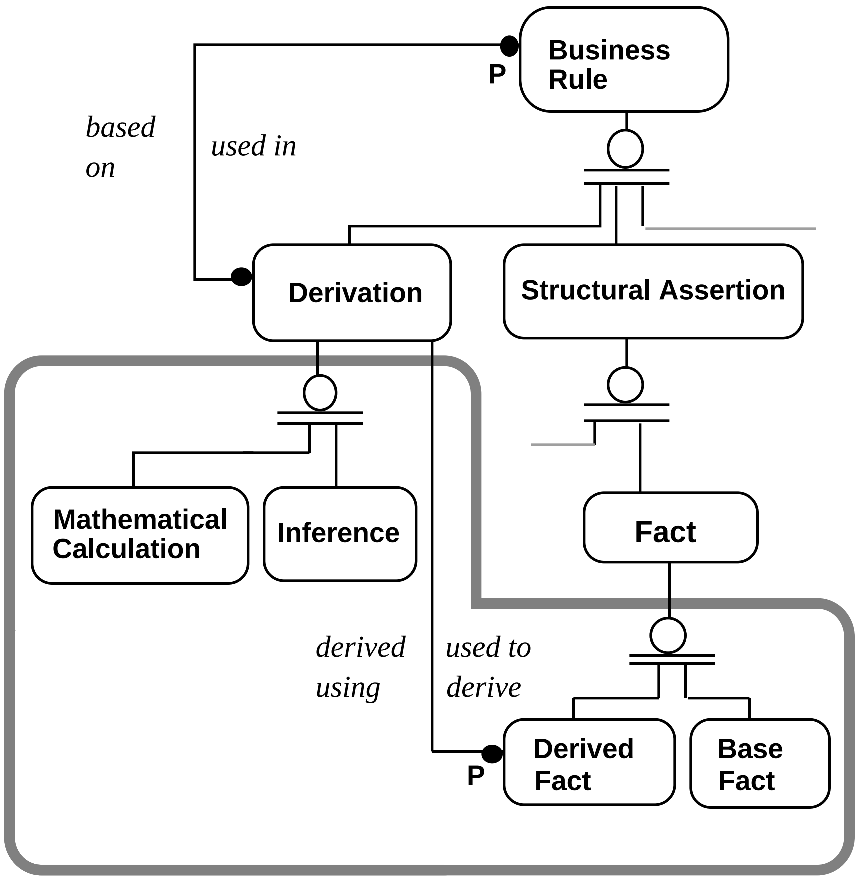


Figure 13: Derived Facts and Derivations

However, a DERIVED FACT may be treated just like any other FACT. Its derivation is not visible when the FACT is referred to.

DERIVATIONS

A DERIVATION is itself a kind of BUSINESS RULE, and it may be either a MATHEMATICAL CALCULATION (such as 'charge rate multiplied by hours') or an INFERENCE (such as the fact that the TIME SHEET ENTRY'S 'charge rate' is in fact the corresponding PERSON'S 'charge rate.' A MATHEMATICAL CALCULATION produces a DERIVED FACT according to a specified mathematical algorithm. An INFERENCE produces a DERIVED FACT using logical induction (from particulars) or deduction (from general principles).

Note, by the way, that implicit in the determination of which is a BASE FACT and which is a DERIVED FACT are assumptions about the order in which information comes available. Here we assumed that the 'charge rate' and 'hours' were known before the 'total cost.' In a different environment, the reverse could be true. It is the job of the analyst to consider these factors and make this judgment.

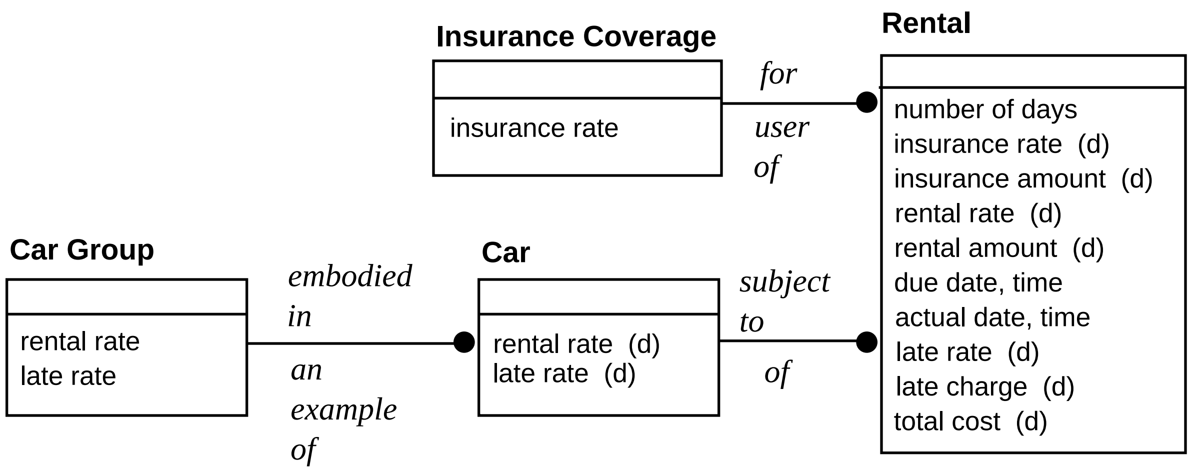


Figure 14: Derived Facts

An example of how DERIVED FACTS can appear on an entity/relationship model is shown in Figure 14. BASE FACTS shown include the presence of 'insurance rate' as an attribute of 'Insurance Coverage,' 'rental rate' as an attribute of 'Car Group,' and so forth.

For the car rental business, the DERIVATIONS of DERIVED FACTS can be specified as follows:

- "The 'Insurance rate' in RENTAL is ***inferred from*** the 'Insurance rate' of the INSURANCE COVERAGE *of* that the RENTAL is the *user of*, through a many-to-one relationship."
- "The 'Rental rate' in RENTAL is ***inferred from*** the 'Rental rate' of the CAR of that the RENTAL is *of*, through a many-to-one relationship. This in turn is inferred from the 'Rental rate' of the CAR GROUP that the CAR is *an example of*."
- "The 'Late rate' in RENTAL is ***inferred from*** the 'Late rate' of the CAR of that the RENTAL is *of*, through a many-to-one relationship. This in turn is inferred from the 'Late rate' of the CAR GROUP that the CAR is *an example of*."
- "The 'Insurance amount' in RENTAL is ***calculated from*** the 'Insurance rate' multiplied by the 'number of days.'"
- "The 'Rental amount' in RENTAL is ***calculated from*** the 'Rental rate' multiplied by the 'number of days.'"
- "The 'Late charge' in RENTAL is ***calculated from*** the difference between the 'Due date, time' and the 'Actual date, time' multiplied by the 'Late rate.'"
- "The 'Total cost' of the RENTAL is ***calculated from*** the sum of 'Insurance amount,' 'Rental amount,' and 'Late charge.'

In each case, the formula involved is an occurrence of DERIVATION which is *based on* one or more BUSINESS RULES, and *used to calculate* one DERIVED FACT.

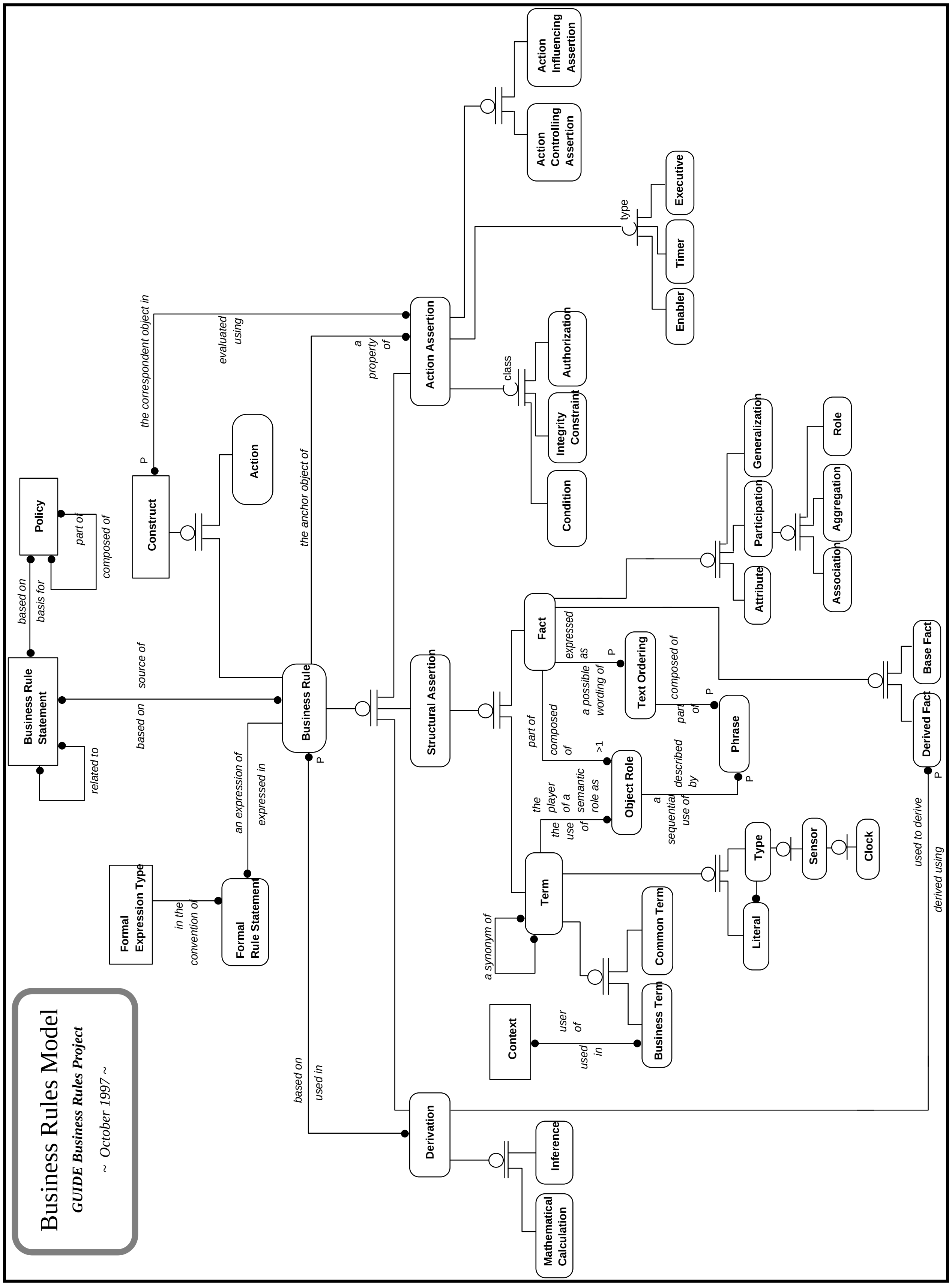
DEFINITIONS

The following summarizes the definitions of the concepts relating to *Derivations*.

Definitions	
Base Fact	a FACT that is a given in the world and is remembered (stored) in the system.
Derivation	an algorithm used to compute or infer a DERIVED FACT.
Derived Fact	a FACT whose value is created by an inference or a mathematical calculation from TERMs, FACTs, other DERIVATIONs, or ACTION ASSERTIONs.
Inference	a DERIVATION that produces a DERIVED FACT using logical induction (from particulars) or deduction (from general principles).
Mathematical Calculation	a DERIVATION that produces a DERIVED FACT according to a specified mathematical algorithm.

Table 4: Derivation Definitions

Appendix A
The Complete Business Rules Model


Business Rules Model
GUIDE Business Rules Project
~ October 1997 ~
Policy
part of
composed of
based on
basis for
Business Rule Statement
related to
source of
based on
Construct
P
the correspondent object in
evaluated using
Action
Business Rule
P
the anchor object of
a property of
Action Assertion
Action Controlling Assertion
Action Influencing Assertion
type
Executive
Timer
Enabler
class
Authorization
Integrity Constraint
Condition
Structural Assertion
Fact
expressed as
P
a possible wording of
Text Ordering
composed of
P
part of
Phrase
P
described by
a sequential use of
Object Role
>1
composed of
part of
the player of a semantic role as
the use of
Generalization
Participation
Attribute
Role
Aggregation
Association
Base Fact
Derived Fact
P
used to derive
derived using
Term
a synonym of
Type
Sensor
Clock
Literal
Common Term
Business Term
Context
user of
used in
Formal Expression Type
in the convention of
Formal Rule Statement
an expression of
expressed in
based on
used in
Derivation
Inference
Mathematical Calculation

Appendix B

How to Read Entity/Relationship Models

Appendix B
How to Read Entity/Relationship Models

The entity/relationship models in this document have been portrayed using the IDEF1X notation, with a specific convention for relationship names. Each relationship, in each direction, is intended to be read as a normal English sentence in the form:

> Each <entity 1> {must be | may be} <relationship> {one and only one | one or more} <entity 2>.

Both relationship names appear next to the relationship line, and they are intended to be read in a clockwise direction. The relationship symbols are as follows:

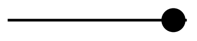
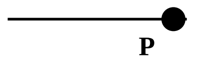
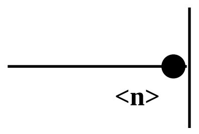
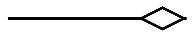
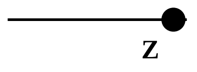
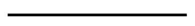
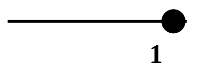
may be <relationship> one or more <entity>s	
must be <relationship> one or more <entity>s	
must be <relationship> (some specific number <n>) <entity>s	
may be <relationship> one and only one <entity>	
may be <relationship> one and only one <entity> (Alternate form)	
must be <relationship> one and only one <entity>	
must be <relationship> one and only one <entity> (Alternate form)	

Figure B-1: Relationship Symbols

For example, the model in Figure B-2 contains the following two sentences:

· Each TEXT ORDERING must be *a possible wording of* one or more FACTS.

· Each FACT must be *expressed as* one and only one TEXT ORDERING.

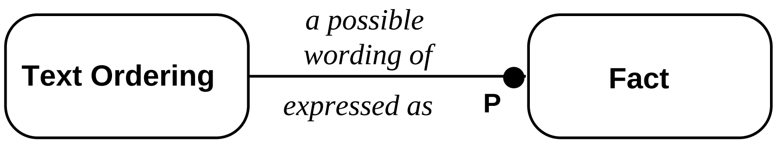


Figure B-2: Reading Fact Sentences

In the drawings, round cornered rectangles represent either subtypes or associative (intersect) entities.

A subtype entity is a set of occurrences of a supertype entity. All attributes of and relationships with a supertype are 'inherited' by the subtype, but each subtype may have its own relationships and attributes unshared with other subtypes. In the IDEF1X notation, an entity may have more than one 'cluster' of subtypes. An occurrence of one subtype may not also be an occurrence of any other subtype in the same cluster.

Where all occurrences of the supertype are also occurrences of one of the subtypes in a cluster (that is, the cluster is described completely), the symbol shown in Figure B-3 is used.

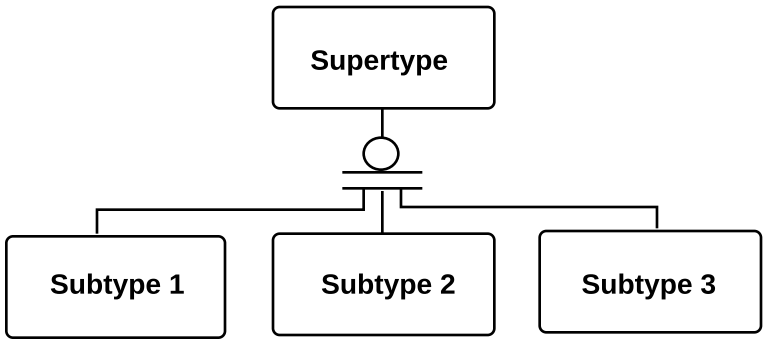


Figure B-3: A Complete (Total) Subtype Cluster

If the set of subtypes do not exhaustively describe all occurrences of the supertype that are in a cluster, the symbol is as shown in Figure B-4.

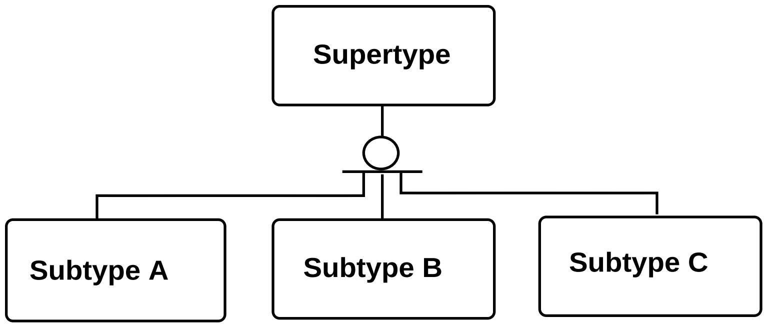


Figure B-4: An Incomplete (Partial) Subtype Cluster

In this document, the model may show a complete cluster, but a view being described in a particular section may not show all the subtypes. In this case, the incomplete view is shown with a dotted line to the side (as illustrated in Figure B-5) to indicate that not all subtypes are shown in the view.

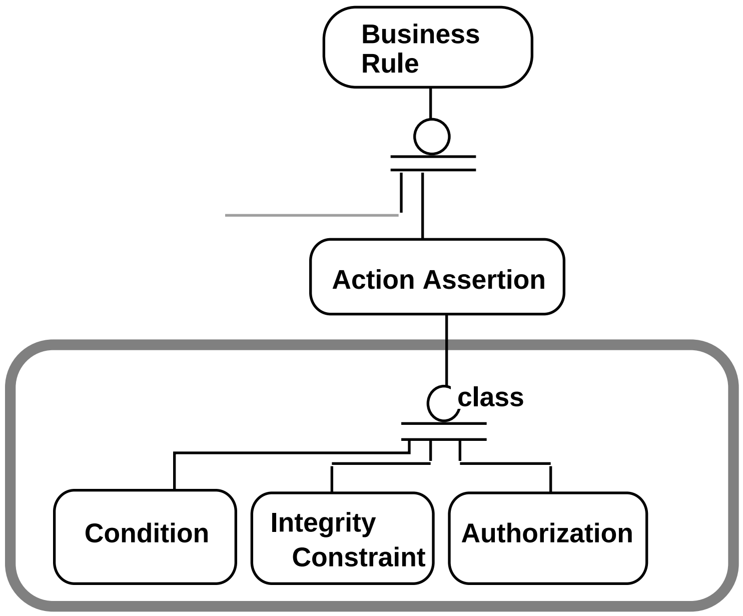


Figure B-5: Convention for Omissions from a View Model

Appendix C

An Extended List of Action Assertion Types

Appendix C
An Extended List of Action Assertion Types

In the main part of the text, rule types ENABLER, EXECUTIVE, and TIMER were discussed. These are in fact but three of a much larger number of rule types. Some possible additional types of ACTION ASSERTION are presented here.[12]

- ***Instance verifiers*** — These rules pertain to individual instances or occurrences of object classes.
    - ***Mandatory*** — something must occur, or a test for its existence.
    - ***Limited*** — constrains the number of instances of an object class that may exist.
    - ***Restricted*** — special case of 'mandatory,' involving recursive structures.
    - ***Pre-existing*** — requires occurrence of correspondent object class to exist before anchor object class, and still exists, or tests for that condition.
    - ***Antecedent*** — requires occurrence of correspondent object class to exist before anchor object class, *whether or not it still exists,* or tests for that condition.
- ***Type verifiers*** — These rules control the creation of multiple instances in various object classes.
    - ***Mutual*** — requires instances of correspondent objects to exist simultaneously, or tests for that condition.
    - ***Mutually exclusive*** — requires that no more than one correspondent object exist simultaneously, or tests for that condition.
    - ***Mutually dependent*** — requires that either one instance of every correspondent object class exists, or that no instances of any correspondent object class exist, or tests for that condition.
    - ***Mutually prohibited*** — requires that at least one of the correspondent object classes has no instances, or tests for that condition.
- ***Sequence verifiers*** — These rules control changes in state. If an object class may be in multiple states (often portrayed on an e/r diagram as subtypes), then these rules determine the sequence in which instances of that object class may assume those states.
    - ***Initializing*** — requires that an instance of the supertype object class must be defined first in a specified state, or tests for that condition..
    - ***Forward*** — requires that once an instance of the supertype object class has attained a state, it cannot move to one lower in the defined sequence, or tests for that condition.

12 Ronald G. Ross, *op. cit.*

  - ***Progressive* —** requires that once an instance of the supertype object class has attained a state, it can only move to the next higher state in the defined sequence, or tests for that condition.
  - ***Retrogressive* —** requires that once an instance of the supertype object class has attained a state, it can only move to the next lower state in the defined sequence, or tests for that condition.
  - ***Re-initializing* —** requires that, if an instance of the supertype object class moves to a lower state in the defined sequence, it must be reset to the initial state, before it can be advanced, or tests for that condition.
  - ***Cyclical* —** requires that an instance of the supertype object class be advanced to the highest state in the sequence defined for this rule, before it can be moved to a lower state, and that the instance must be moved back to the lowest state in the sequence defined for this rule, before it can be moved to a higher state, or tests to see if an instance has been in both the highest and lowest state.
- ***Position selectors* —** These rules pertain to the position of a value (either of an attribute, or of the identifying attribute of an object class), either in a value sequence or a chronological sequence.
  - ***Positioned, lowest and highest*** — for value sequences, requires that the value of an instance of the anchor object class must be higher or lower than the values of all instances of the correspondent object class, or tests for that condition..
  - ***Chronological, oldest and newest* —** for chronological sequences, requires that the value of an instance of the anchor object class must be older or newer than the values of all instances of the correspondent object class or tests for that condition.
- ***Functional evaluators*** — These rules control the sequence in which instances of an object class are defined.
  - ***Unique*** — requires that no two instances of the anchor object class may possess the same values for the instance(s) of the correspondent object, or tests for that condition.
  - ***Fluctuating*** — requires that no *successive* instances of the anchor object class may possess the same values for the instance(s) of the correspondent object, or tests for that condition.
  - ***Ascending* —** requires that instances of a correspondent object class must increase in value for all successive instances of its anchor object, or tests for that condition.
  - ***Descending*** — requires that instances of a correspondent object class must decrease in value for all successive instances of its anchor object, or tests for that condition.
  - ***Non-renewable*** — requires that any given value of the correspondent object class, if used more than once, may be used only in strictly successive instances of the anchor object.
  - ***Patterned*** — requires that successive instances of the anchor object class be assigned in a specified sequence, or tests for that condition.

  - ***Functional*** — requires the value of any instance of the correspondent object class to be produced as a function of the relative position of the given instance of the anchor object class in the succession of all its peers.

- ***Comparative evaluators*** — These rules describe comparisons between pairs of instances of object classes. The comparisons may be 'equal to,' 'not-equal-to,' 'greater-than,' 'greater-than-or-equal-to,' 'less-than,' or 'less-than-or-equal-to,' and so forth.

- ***Calculators*** — These rules test the result of a standard computation involving values of the correspondent object class(es). Any standard computation may be the subject of one of these rules. Among the possibilities are: 'Summed,' 'Subtracted,' 'Maximum,' 'median,' and so forth.

  - ***Type 1****:* The first type of calculator rule always indicates that an attribute is the anchor object. The result of the computation indicated for instance(s) of the rule's correspondent object class(es) is tested against the current value of this attribute type.

  - ***Type 2****:* The second type of calculator rule always indicates a data object class as its anchor object class. This means that no calculations are actually performed on the anchor object class. For this reason, all type 2 calculators can only be conditions, never integrity constraints.

- ***Update controllers*** — These rules prescribe whether updates to a database may occur.

  - ***Frozen*** — requires that instance(s) of the correspondent object may not be created, deleted and/or modified while an instance of the anchor object exists, or tests for that condition.

  - ***Frozen-to-users*** — like *Frozen*, requires that instances of a correspondent object must increase in value for all successive instances of its anchor object, or tests for that condition, but this applies only to user-specified updates, not to any created automatically.

  - ***Enabled*** — requires existence for every instance of the correspondent object class, or tests for that condition. That is, if an instance of the anchor object exists, then the correspondent object is enabled, or the condition tests to see if the correspondent object is enabled.

  - ***Enabled with reversal*** — like *Enabled*, requires existence for every instance of the correspondent object class, or tests for that condition. However, unlike *enabled*, if the rule's anchor object is deleted, the state of the correspondent objects is reversed.

- ***Timing controllers*** — These rules prescribe tests for the length of time that instances of correspondent objects have (or have not) existed.

  - ***Timed*** — requires a timing threshold for instances of the correspondent object class(es), or tests for that threshold.

Appendix D

Case Study: EU-Rent Car Rentals

Appendix D
EU-Rent Car Rentals

Examples in this document are based on the following Case Study. This case study was developed by Model Systems, Ltd., along with several other organizations, and has been used by other organizations. The material below may be copied and used freely, if its source is clearly acknowledged.

EU-RENT CAR RENTALS

EU-Rent is a car rental company owned by EU-Corporation. It is one of three businesses ~ the other two being hotels and an airline ~ that each has its own business and IT systems, but with a shared customer base. Many of the car rental customers also fly with EU-Fly and stay at EU-Stay hotels.

EU-RENT BUSINESS

EU-Rent has 1000 branches in towns in several countries. At each branch cars, classified by car group, are available for rental. Each branch has a manager and booking clerks who handle rentals.

Rentals

Most rentals are by advance reservation; the rental period and the car group are specified at the time of reservation. EU-Rent will also accept immediate ('walk-in') rentals, if cars are available.

At the end of each day cars are assigned to reservations for the following day. If more cars have been requested than are available in a group at a branch, the branch manager may ask other branches if they have cars they can transfer to him/her.

Returns

Cars rented from one branch of EU-Rent may be returned to a different branch. The renting branch must ensure that the car has been returned to some branch at the end of the rental period. If a car is returned to a branch other than the one that rented it, ownership of the car is assigned to the new branch.

Servicing

EU-Rent also has service depots, each serving several branches. Cars may be booked for maintenance at any time provided that the service depot has capacity on the day in question.

For simplicity, only one booking per car per day is allowed. A rental or service may cover several days.

Customers

A customer can have several reservations but only one car rented at a time. EU-Rent keeps records of customers, their rentals and bad experiences such as late return, problems with payment and damage to cars. This information is used to decide whether to approve a rental.

EU-RENT BUSINESS RULES

External constraints

Each driver authorized to drive the car during a rental must have a valid driver's license.

Each driver authorized to drive the car during a rental must be insured to the level required by the law of each country that may be visited during the rental.

Rented cars must meet local legal requirements for mechanical condition and emissions for each country that may be visited during the rental.

Local tax must be collected (at the drop-off location) on the rental charge.

Rental reservation acceptance

If a rental request does not specify a particular car group or model, the default is group A (the lowest-cost group).

Reservations may be accepted only up to the capacity of the pick-up branch on the pick-up day.

If the customer requesting the rental has been blacklisted, the rental must be refused.

A customer may have multiple future reservations, but may have only one car at any time.

Car allocation for advance reservations

At the end of each working day, cars are allocated to rental requests due for pick-up the following working day. The basic rules are applied within a branch:

- only cars that are physically present in EU-Rent branches may be assigned.
- if a specific model has been requested, a car of that model should be assigned if one is available. Otherwise, a car in the same group as the requested model should be assigned.
- if no specific model has been requested, any car in the requested group may be assigned.
- the end date of the rental must be before any scheduled booking of the assigned car for maintenance or transfer.

- after all assignments within a group have been made, 10% of the group quota for the branch (or all the remaining cars in the group, whichever number is lower) must be reserved for the next day's walk-in rentals. Surplus capacity may be used for upgrades.
- if there are not sufficient cars in a group to meet demand, a one-group free upgrade may be given (i.e., a car of the next higher group may be assigned at the same rental rate) if there is capacity.
- customers in the loyalty incentive scheme have priority for free upgrades.

If demand cannot be satisfied within a branch under the basic rules, one of the 'exception' options may be selected:

- a car may be allocated from the capacity reserved for the next day's walk-ins.
- a 'bumped upgrade' may be made. For example, if a group A car is needed and there is no capacity in group A or B, then a car allocated to a group B reservation may be replaced by a group C car, and the freed-up group B car allocated to the group A reservation.
- a downgrade (a car of a lower group) may be made.
- a car from another branch may be allocated, if there is a suitable car available and there is time to transfer it to the pick-up branch.
- a car due for return the next day may be allocated, if there will be time to prepare it for rental before the scheduled pick-up time.
- a car scheduled for service may be used, provided that the rental would not take the mileage more than 10% over the normal mileage for service.

If demand cannot be satisfied within a branch under the 'exception' rules, one of the 'in extremis' options may be selected:

- pick-up may have to be delayed until a car is returned and prepared.
- a car may have to be rented from a competitor.

Walk-in rentals

The end date of the rental must be before any scheduled booking of the assigned car for maintenance or transfer.

If there are several available cars of the model or group requested, the one with the lowest mileage should be allocated.

Handover

Each driver authorized to drive the car during a rental must be over 25 and have held a driver's license for at least one year.

The credit card used to guarantee a rental must belong to one of the authorized drivers; and this driver must sign the rental contract. Other drivers must sign an ‘additional drivers authorization’ form.

The driver who signs the rental agreement must not currently have a EU-Rent car on rental.

Before releasing the car, a credit reservation equivalent to the estimated rental cost must be made against the guaranteeing credit card.

The car must not be handed over to a driver who appears to be under the influence of alcohol or drugs.

The driver must be physically able to drive the car safely — must not be too tall, too short or too fat; if disabled, must be able to operate the controls.

The car must have been prepared — cleaned, full tank of fuel, oil and water topped up, tires properly inflated.

The car must have been checked for roadworthiness — tire tread depth, brake pedal and hand brake lever travel, lights, exhaust leaks, windscreen wipers.

No-shows

If an assigned car has not been picked up 90 minutes after the scheduled pick-up time, it may be released for walk-in rental, unless the rental has been guaranteed by credit card.

If a rental has been guaranteed by credit card and the car has not been picked up by the end of the scheduled pick-up day, one day’s rental is charged to the credit card and the car is released for use the following day.

Return from rental

At the end of a rental, the customer may pay by cash, or by a credit card other than the one used to guarantee the rental.

If a car is returned to a location other than the agreed drop-off branch, a drop-off penalty is charged.

The car must be checked for wear (brakes, lights, tires, exhaust, wipers etc.) and damage, and repairs scheduled if necessary.

If the car has been damaged during the rental and the customer is liable, the customer’s credit card company must be notified of a pending charge.

Early returns

If a car is returned early, the rental charge is calculated at the rate appropriate to the actual period of rental (e.g., daily rate rather than weekly).

Late returns

If the car is returned late, an hourly charge is made up to 6 hours' delay; after 6 hours a whole day is charged.

A customer may request a rental extension by phone — the extension should be granted unless the car is scheduled for maintenance.

If a car is not returned from rental by the end of the scheduled drop-off day and the customer has not arranged an extension, the customer should be contacted.

If a car is three days overdue and the customer has not arranged an extension, insurance cover lapses and the police must be informed.

Car maintenance & repairs

Each car must be serviced every three months or 10,000 kilometers, whichever occurs first.

If there is a shortage of cars for rental, routine maintenance may be delayed by up to 10% of the time or distance interval (whichever was the basis for scheduling maintenance) to meet rental demand.

Cars needing repairs (other than minor body scratches and dents) must not be used for rentals.

Car purchase and sale

Only cars on the authorized list can be purchased.

Cars are to be sold when they reach one year old or 40,000 kilometers, whichever occurs first.

Car ownership

A branch cannot refuse to accept a drop-off of a EU-Rent car, even if a one-way rental has not been authorised.

When a car is dropped off at a branch other than the pick-up branch, the car's ownership (and, hence, responsibility for it) switches to the drop-off branch when the car is dropped off.

When a transfer of a car is arranged between branches, the car's ownership switches to the 'receiving' branch when the car is picked up.

In each car group, if a branch accumulates cars to take it more than 10% over its quota, it must reduce the number back to within 10% of quota by transferring cars to other branches or selling some cars.

In each car group, if a branch loses cars to take it more than 10% below its quota, it must increase the number back to within 10% of quota by transferring cars from other branches or buying some cars.

Loyalty incentive scheme

To join the loyalty incentive scheme, a customer must have made 4 rentals within a year.

Each paid rental in the scheme (including the 4 qualifying rentals) earns points that may be used to buy 'free rentals.'

Only the basic rental cost of a free rental can be bought with points. Extras, such as insurance, fuel and taxes must be paid by cash or credit card.

A free rental must be booked at least fourteen days before the pick-up date.

Free rentals do not earn points.

Unused points expire three years after the end of the year in which they were earned.

EXAMPLES OF 'RULES FOR RUNNING THE BUSINESS'

(not really the same kind of rules as those above)

Each branch must be set targets for performance — numbers of rentals, utilization of cars, turnover, profit, customer satisfaction, etc.

Where performance requirements conflict (e.g., profit vs. customer satisfaction when a customer requests a reduction in charges after an unsatisfactory rental) heuristics must be provided to guide branch staff.

Performance data must be captured.

If performance targets are not met, control action must be taken. Control action may include:

- changing the resources at branches (e.g., numbers of cars, quotas of cars within each group, number of staff),
- changing responsibilities (e.g., having transfers of cars managed by groups of branches, rather than by negotiation between individual branch managers),
- changing operational guidance (e.g., what proportion of cars should be kept for walk-in rentals), but not external constraints (e.g., legal requirements) or company policies (e.g., rentals must be guaranteed by a credit card, a customer may have only one car at a time).

Appendix E

Concepts Catalog of Model Terms and Definitions

Appendix E
Concepts Catalog of Model Terms and Definitions

This section consolidates the definitions of the business rule concepts documented in the model. The chapter in which the term is discussed in detail is shown on the right.

TERM	*DEFINITION*	*Chapter*
Action	something that executes and may change the state of one or more instances of one or more TYPEs. An ACTION has a protocol and one or more methods that implement it. An ACTION cannot be constrained; only TYPEs (things which have persistent instances) can be constrained. The enabling and execution of an ACTION can be controlled through rules . The ACTION is permitted to proceed once/if conditions are satisfied.	**5**
Action Assertion	a statement that concerns some dynamic aspect of the business. It specifies constraints on the results that actions can produce.	**5**
Action Controlling Assertion	an ACTION ASSERTION that describes what must or must not be (or happen).	**5**
Action Influencing Assertion	an ACTION ASSERTION that describes what should or should not be (or happen).	**5**
Aggregation	a specialization of PARTICIPATION that expresses an 'is part of / is composed of' FACT. The TERMS in the FACT describe the component types of the whole.	**4**
Association	a specialization of PARTICIPATION that expresses an 'is associated with' FACT between two or more TYPEs. Ideally, an ASSOCIATION is described using a verb phrase that expresses the particular nature of the association. It represents any type of PARTICIPATION (relationship) other than AGGREGATION or ROLE.	**4**
Attribute	a specialization of FACT that expresses a 'has property of' relationship between TERMs, specifically an association between an entity type and a domain/abstract data type. For example, "a customer has a name."	**4**
Authorization	an assertion that a specific prerogative or privilege has been defined with respect to one or more CONSTRUCTS. An AUTHORIZATION is an assertion represented the predicate: "(Only) *x* may do *y*," where *x* typically is a user and *y* is an action that may be executed.	**5**
Base Fact	a FACT that is a given in the world and is remembered (stored) in the system.	**6**
Base Fact	a FACT that is not a DERIVED FACT.	**4**

Business Rule	a statement that defines or constrains some aspect of the business. This must be either a term or fact (described below as a STRUCTURAL ASSERTION), a constraint (described below as an ACTION ASSERTION), or a DERIVATION. It is 'atomic' in that it cannot be broken down or decomposed further into more detailed business rules. If reduced any further, there would be loss of important information about the business.	**3**
Business Rule Statement	a declarative statement of structure or constraint which the business places upon itself or has placed upon it.	**3**
Business Term	a word or phrase that has a specific meaning for a business in some designated CONTEXT.	**4**
Clock	a special kind of SENSOR whose value is 'real-world time.' A CLOCK must have exactly one value, which is 'current time.'	**4**
Common Term	a word or phrase in everyday language using its commonly-accepted meaning. Specifically, common terms are part of the basic vocabulary, for example, 'year,' 'calendar,' etc., and are taken as axiomatic to avoid writing circular definitions.	**4**
Condition	an assertion that *if* something is true, another business rule will apply. A CONDITION can be thought of as a 'test' ~ if true, it may be the basis for enforcing or testing other ACTION ASSERTIONs.	**5**
Construct	a generalization that represents either a BUSINESS RULE or an ACTION.	**5**
Context	an environment where shared BUSINESS TERMs are used with an agreed-to meaning.	**4**
Derivation	an algorithm used to compute or infer a DERIVED FACT.	**6**
Derived Fact	a FACT whose value is computed or inferred from other FACTS via a specified DERIVATION.	**4**
Derived Fact	a FACT whose value is created by an inference or a mathematical calculation from TERMs, FACTs, other DERIVATIONs, or ACTION ASSERTIONs.	**6**
Fact	an associating of two or more TERMS. It expresses a potential for association ('can be' or 'may be') rather than expressing a 'must be' association.	**4**
Formal Expression Type	one of the formal grammars for representing BUSINESS RULES.	**3**
Formal Rule Statement	an expression of a BUSINESS RULE in a specific formal grammar.	**3**
Generalization	a specialization of FACT in which one TERM (a TYPE) describes a subset of occurrences of another TERM (also a TYPE).	**4**
Inference	a DERIVATION that produces a DERIVED FACT using logical induction (from particulars) or deduction (from general principles).	**6**

Integrity Constraint	an assertion that must always be true.	**5**
Literal	a kind of TERM that reflects a specific value or instance of a TYPE.	**4**
Mathematical Calculation	a DERIVATION that produces a DERIVED FACT according to a specified mathematical algorithm.	**6**
Object Role	the semantic role that a TERM plays in a FACT.	**4**
Participation	any association between TERMS other than ATTRIBUTE or GENERALIZATION. (This would typically appear as a relationship in an entity/relationship diagram.)	**4**
Phrase	one component of a TEXT ORDERING that reflects the usage of an OBJECT ROLE in a specific position of the TEXT ORDERING, identifying its syntactic role and providing its portion of the text (with marker).	**4**
Policy	a general statement of direction for an enterprise.	**3**
Role	a statement of the way in which one TERM may serve as an actor (another TERM) through its interactions with its environment. For example, "a customer may be *a* buyer *in* a contract." *(or, a* seller *in, the* recipient *of*, etc.)	**4**
Sensor	a special kind of TYPE whose value is asserted by some mechanism or device whose inner workings are unknown (or uninteresting to the identified scope). Its value cannot be altered directly. A SENSOR detects and reports constantly changing values from the outside world, such as the passage of time, a temperature reading, or some other value.	**4**
Structural Assertion	a statement that something of importance to the business either exists as a concept of interest or exists in relationship to another thing of interest. It details a specific, static aspect of the business, expressing things to be known or how known things fit together.	**4**
Term	a word or phrase used by the business.	**4**
Text Ordering	the portrayal of one possible wording of a FACT. It is an aggregation of its constituent PHRASES.	**4**
Type	a kind of TERM that is a named abstraction of a set of instances or values.	**4**

Appendix F

The Model Graphic in UML Notation

Appendix F
The Model Graphic in UML Notation

As explained in Appendix B, the entity/relationship models in the body of this document have been portrayed using the IDEF1X notation. In 1999, we translated the representation into the Unified Modeling Language (UML) notation. Since numerous references are available to explain the graphics[13] the notation will not be explained here.

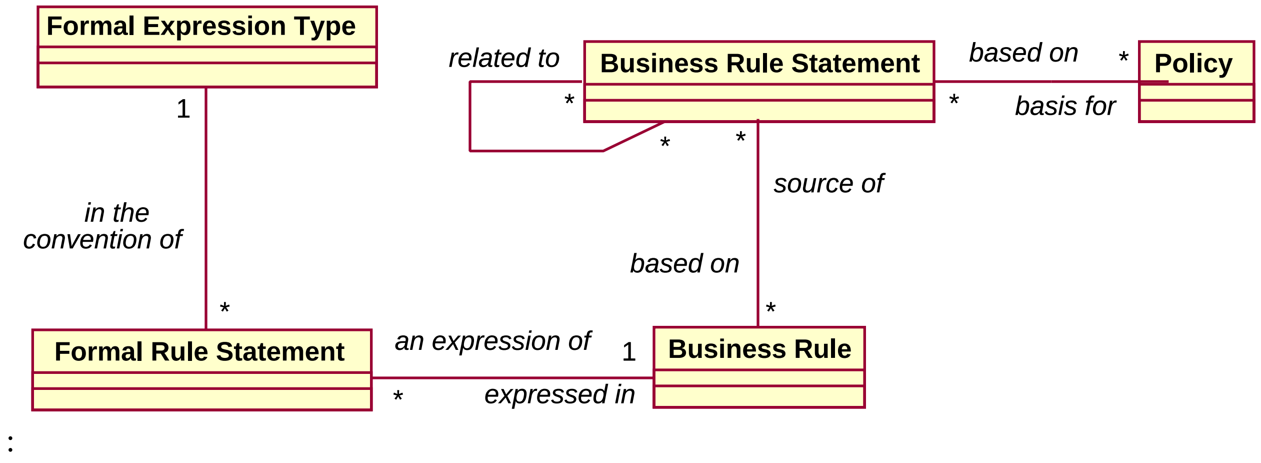


Figure F-1: The Origin of Business Rules
Corresponds to Figure 3 [p. 10]

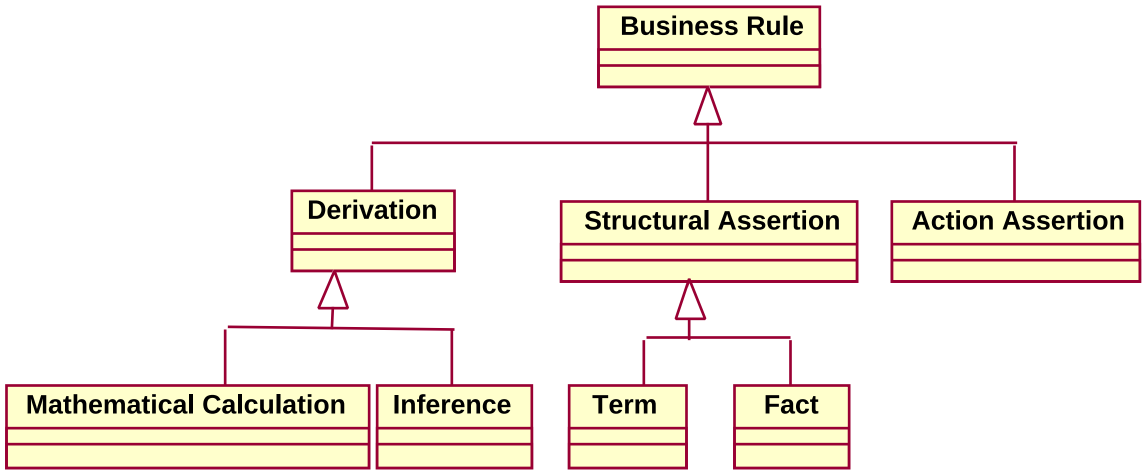


Figure F-2: Business Rule Types
Corresponds to Figures 4 [p. 13] and 13 [p. 31]

13 For example, see: Martin Fowler, UML Distilled Second Edition, Addison Wesley, (c)2000, or Grady Booch (et al), The Unified Modeling Language User Guide, Addison-Wesley, (c)1999.

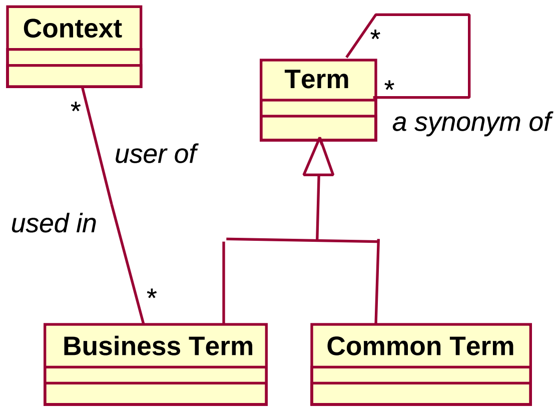


Figure F-3: Terms
Corresponds to Figure 5 [p. 15]

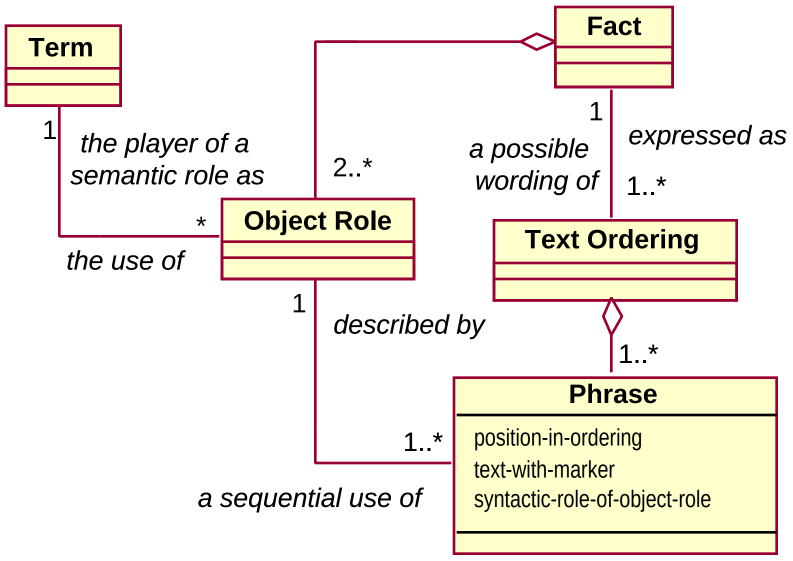


Figure F-4: Facts
Corresponds to Figure 5 [p. 15]

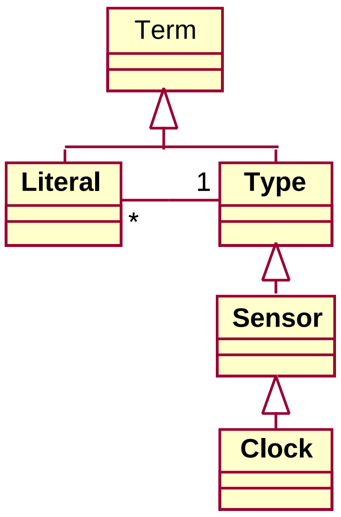


Figure F-5: Kinds of Term
Corresponds to Figure 7 [p. 18]

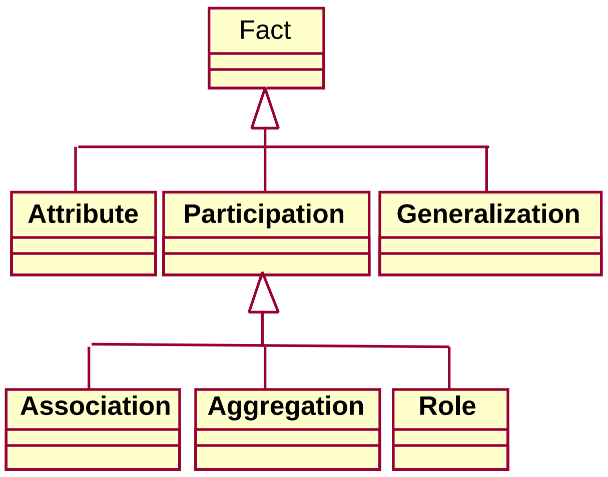


Figure F-6: Kinds of Fact
Corresponds to Figure 8 [p. 19]

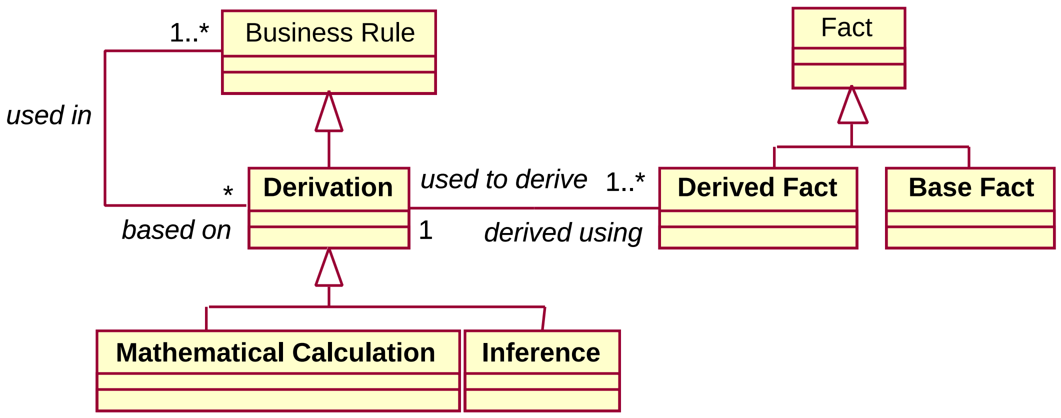


Figure F-7: Kinds of Derivation
Corresponds to Figures 8 [p. 19] and 13 [p. 31]

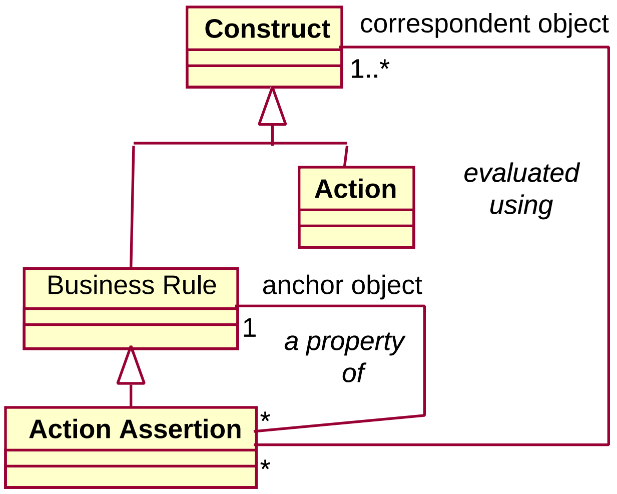


Figure F-8: Action Assertions
Corresponds to Figure 9 [p. 25]

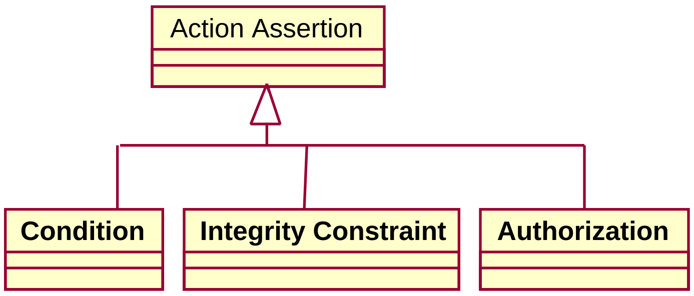


Figure F-9: Types of Action Assertion
Corresponds to Figure 10 [p. 27]

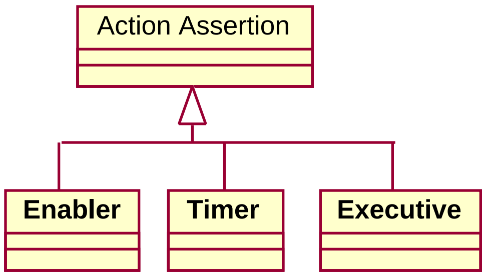


Figure F-10: Classes of Action Assertion
Corresponds to Figure 11 [p. 28]

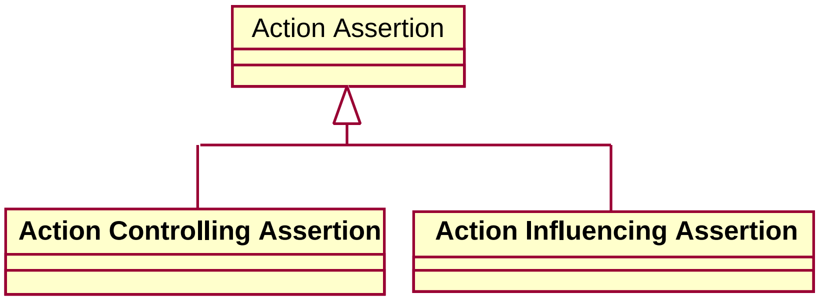


Figure F-11: Action Controlling vs. Action Influencing Assertions
Corresponds to Figure 12 [p. 29]