

Identification Pattern

Pavel Hruby

Adapted from the book by Hruby, P. Kiehn J. Scheller C.V.: Model-Driven Design Using Business Patterns, Springer, 2006.



Barcode is a machine readable strip for automatic identification of items, by means of printed bars of different widths

Context

REA (Resources Events, Agents) modeling framework determines the structure of a business application, which conforms to the fundamental laws of the business domain. To build a useful business application, this structure is only one of the things an application developer has to determine. Users of business applications usually require additional functionality, such as serial numbers, accounts, price calculations, and conversions between units of measure. This functionality is essential in some applications, but it might not be required in others. All depends on the users of a business application, actual configuration of an application, and the common practices in their businesses.

An example of the functionality often required in business applications, but not specified by REA, is naming and numbering of business entities. People refer to real or imaginary things by their names. We name things to identify them, so we can refer to them by their names and not just point to them and say "this!". By naming, we give things identities, but in real life they are not often unique. Many things have more than one name, and sometimes a single name can refer to different things, which is fine as long as everyone who uses that name knows what thing it refers to. In business,

people use serial numbers, production numbers, social security numbers, and names.

Problem

How do we specify the identity of things represented by REA entities?

Forces (i.e. Constraints and Requirements)

The solution needs to balance the following forces:

- An identity is a given feature; it is not an intrinsic part of the objects and things. Therefore, an REA application model must specify whether there is a business reason requiring REA entities to have a distinct identity, and how that identity is modeled. We could omit modeling identity of an entity, but then we could distinguish different instances of this entity only by the values of their attributes.
- Users of business applications do not necessarily require that all REA entities have an explicit identifier. For example, users of business applications might not be interested in managing the identifiers of sales order lines.
- Some identifiers are unique in the universe, such as the GUID (Global Unique Identifier); some are not unique, such as the first name and last name of a person. Some identifiers are unique within a certain group, such as a serial number, which is unique within the group of entities that belong to the same number series.
- There are specific rules on how to construct identifiers. For example, the ISBN (International Standard Book Number) or the numbers of major credit cards are constructed in a way that enables verifying, using a simple calculation algorithm, whether the number is valid.

Solution

The *Identification Pattern* can be used in situations in which application developers want to specify the identity of REA entities. The *Identification Pattern* consists of two related elements, the *Identifier* and *Identifier Setup*.

The *Identifier* element is contained by any REA entity that needs to be identifiable, for example, should have a name or number. The *Identifier* element contains the attribute *ID String*, which provides an identity to each REA entity instance. It is common that an REA entity contains more than

one *Identifier*; for example, an employee might be identified both by his name and his social security number.

The *Identifier Setup* is usually contained on a group of REA entities that share the same *ID String* format and rules for creating or validating the *ID String*, for example, on a group of REA entities that belongs to the same number series. As not all REA entities that need to be identifiable are parts of some group, the *Identifier Setup* is often omitted from the model, or is implicit in a software application, for example, as a system table.

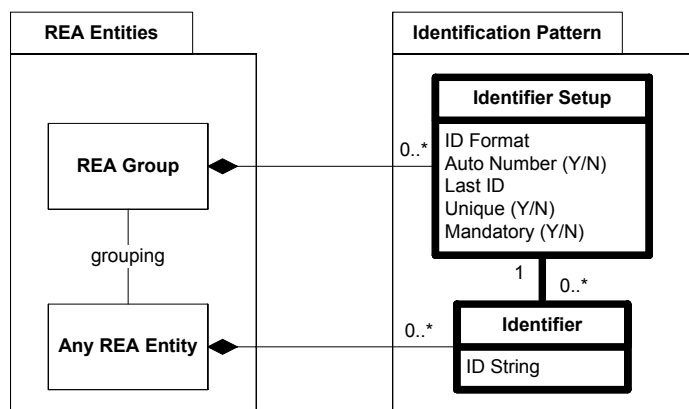


Fig. 1. Sketch of the identification pattern

The *Identifier Setup* contains the configuration parameters, which are used by business logic to validate and construct the *ID Strings*.

The *ID Format* determines how the *ID Strings* are created (most business applications often require combinations of letters and numbers). The *ID Format* can also be used by business logic for validating the *ID Strings* entered manually by the users of the business application.

The *AutoNumber* is a Boolean function that can be set on or off to indicate whether the *ID String* can be automatically generated or not; automatically generated number is often referred to as a number series. If the *AutoNumber* attribute is set to 'yes', the attribute *Last ID* stores the last used identification string in the series.

Unique is a Boolean function that can be set on or off to indicate whether or not the *Identifier* is required to be unique within the REA group.

Mandatory is a Boolean function that can be set on or off to indicate whether the *ID String* must have a value at runtime or can be undefined.

The rules for creating new *ID Strings* can vary from simple series with linear increments to rules that allow for validity checks of the identification strings, such as credit card numbers. Legislation in some countries requires that numbers of some business documents consecutive, without gaps, which imposes an extra requirement on how the number is constructed. If an REA entity has been created by omission and deleted after another REA entity of the same series has been created, the business logic must be able to identify the gap in the series and reuse the number of the deleted document.

Examples

The identification pattern is present in almost all software business applications in various application areas.

The *Social Security Number (SSN)*, see Fig. 2, of an employee is an identification that is not an *auto-number*, is *unique*, and is *not mandatory*. The *Identifier Setup* has the name *Social Security Number Setup*, and contains an *ID Format* that determines how the social security number is constructed or verified. The *Identifier* has the name *Social Security Number*, and its *ID String* at runtime contains the social security number.

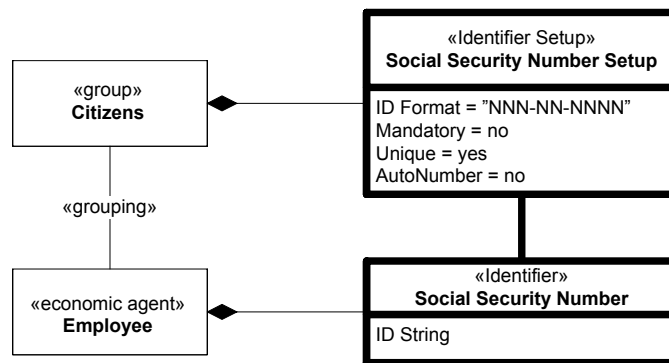


Fig. 2. Social Security Number

Sales Order Number, see Fig. 3, is an identification that is an *auto-number*, is *unique*, and is *mandatory*. As the *Sales Order Number* is an auto-number, the *Sales Order Number Setup* element contains the attribute *Last ID*.

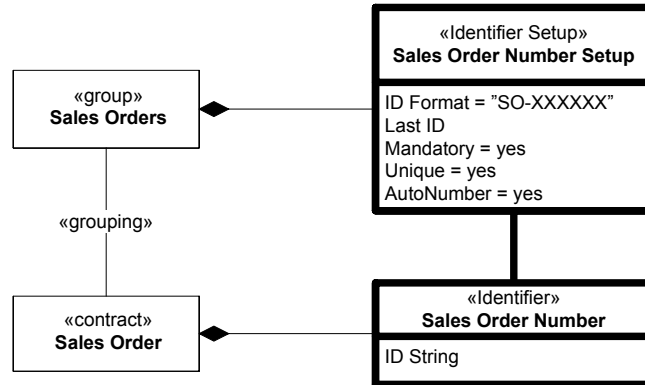


Fig. 3. Sales order number

Product Number, see Fig. 4, is an identification that is an *auto-number*, is *unique*, and is *mandatory*. The configuration of *Product Number* is similar to that of *Sales Order Number* in Fig. 3; *Product Number* is configured on the economic resource *Product*, and *Product Number Setup* is configured on a group of *Products* that belong to the same series.

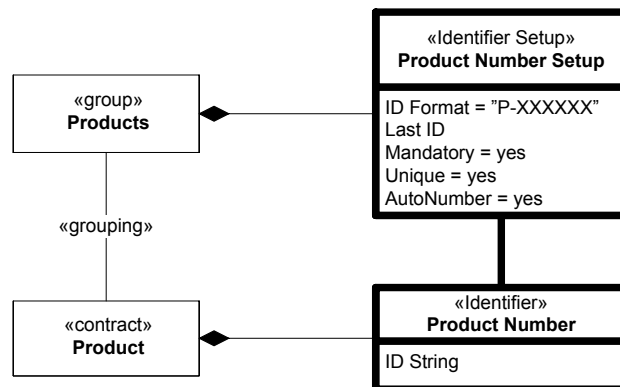


Fig. 4. Product number

Employee Name, see Fig. 5, is an identification that is not an *auto-number*, is not *unique*, but is *mandatory*. *First Name*, *Middle Name*, *Last Name* and *Nickname* share the same *ID Format* specified by *Employee Name Setup*.

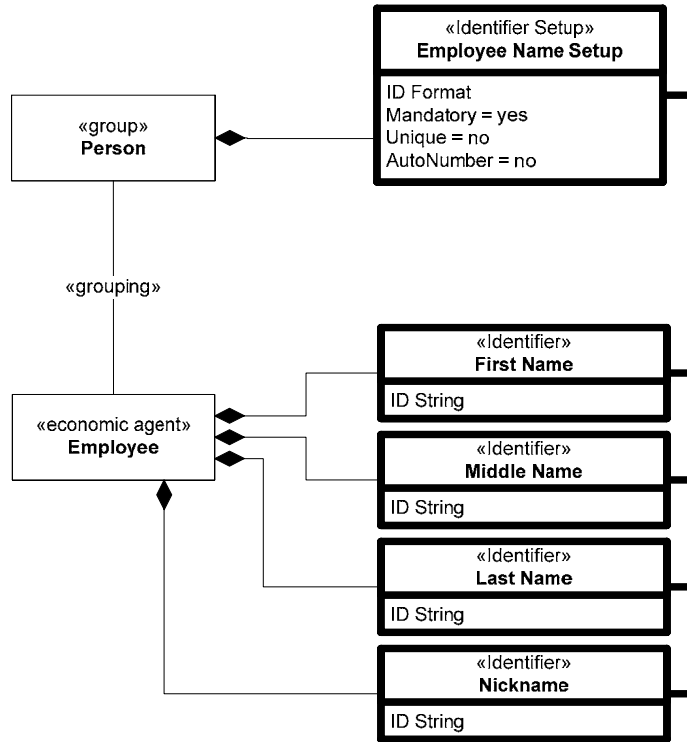


Fig. 5. Employee name

Resulting Context

Sometimes, users of business applications use *phone number*, *e-mail address*, or *Internet address* as identifiers of their trading partners. These numbers and addresses have multiple and different semantics. Phone number can also be used as a contact address, e-mail address as a contact address and destination location (for sending electronic documents and files), and Internet address as a description of the trading partner. In such cases, different patterns, such as *Address*, *Notification*, and *Description*, will contain or refer to the same data (both *Identification* and *Notification* will contain or refer to the same phone number).

There are several international standards specifying Identification Strings and ID formats for economic resources and economic agents in

various lines of business. Examples are European Article Numbering (EAN) for industrial products, International Standard Book Number (ISBN) for books, International Standard Serial Number (ISSN) for periodicals, and International Standard Music Number (ISMN) for printed music publications. For companies, the Data Universal Numbering System (DUNS) is used. References to these standards can be found, for example, in (Arlow, Neustadt 2003).

Acknowledgements

The identification pattern has been discovered by Christian Vibe Scheller and first time described by Jamaica team at Navision in the period 2000-2002.

References

- Arlow J, Neustadt I (2003) Enterprise Patterns and MDA: Building Better Software with Archetype Patterns and UML, Addison-Wesley Professional
- Geerts GL, McCarthy WE (2000b) Augmented Intensional Reasoning in Knowledge-Based Accounting Systems. *Journal of Information Systems*, Volume 14, No. 2, 2000, pp. 127-150.
- Geerts GL, McCarthy WE (2002) An Ontological Analysis of the Primitives of the Extended REA Enterprise Information Architecture” at <http://www.msu.edu/user/mccarth4/>
- Hruby P, Kiehn J, Scheller CV (2006) Model-Driven Design Using Business Patterns, Springer