UNIVERSAL data models LLC

What are Universal Patterns for Data Modeling?

By Len Silverston and Paul Agnew
Published: February 11, 2009

Much of the material in this article is excerpted with permission from the newly released book, "The Data Model Resource Book, Volume 3: *Universal Patterns for Data Modeling*." by Len Silverston and Paul Agnew, published by Wiley Computer Publishing, and released Jan. 2009.

We have a useful rule of thumb that seems to apply to most data models: One-third of a data model usually consists of common constructs that are applicable to most organizations, one-third of the data model is usually industry-specific, and one-third of the model is specific to an organization. What we have also found in our experience with decades of data modeling is that there are very common patterns, what we call universal patterns, that apply to well *over 50 percent of most data model constructs* and that can be reused. For example, a status for an order works in the same way as a status for a shipment. The classification of product or person follows the same pattern, regardless of the fact that one classification deals with products and the other is about people. It is this 50 percent that we address in this article and in our new book, Volume 3 of *The Data Model Resource Book: Universal Patterns for Data Modeling.*

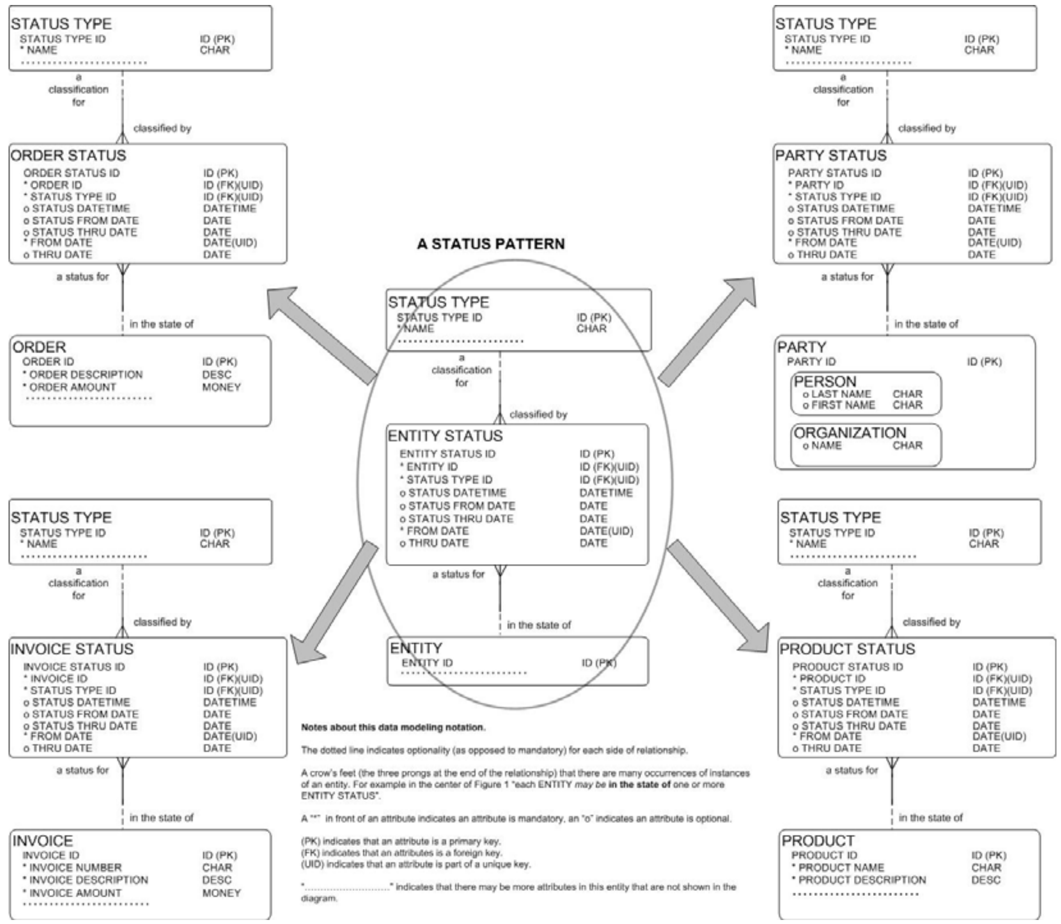
Many of you will have encountered different reusable data models. For example, the first two volumes of *The Data Model Resource Book* and David Hay's excellent book *Data Model Patterns: Conventions of Thought* contain reusable data models for very common data modeling requirements such as how to model data about parties, products, orders/contracts, bill of materials, health care visits, and so on. Also, there are various reusable data models designed for different industries that are available. Many people think of these as providing "patterns" in the context of data modeling; however, these reusable data models contain something very different than what we describe in this article and in *Data Model Resource Book: Universal Patterns for Data Modeling*. In this article, we have a very different meaning when we use the term *pattern*. We make a distinction between a reusable model for a specific application (reusable models that are covered in these other books and products) and the underlying, core templates that are independent of any particular application (which is what we are addressing here).

One way to think of Universal Patterns is in terms of furniture; first think of the design for a dovetail joint, an interlocking technique used to make all sorts of furniture (this is akin to a Universal Pattern), then think of the design for a full set of table and chairs (this is akin to a "Universal Data Model"), and you have an idea of how the concepts relate. Universal Patterns for Data Modeling provide the underlying structural themes so that the modelers can reuse these to build any model, even ones that are very unique!

Universal Patterns for Data Modeling are analogous to the blueprints engineers use for building bridges. An engineer has a basic blueprint for building any type of suspension bridge. Every time an engineer has to build a particular suspension bridge, that engineer doesn't try to come up with a new solution; he or she uses the existing design pattern. The facing on the bridge may be different, but many of the underlying structures are the same. For example, Akashi-Kaikyo Bridge in Tokyo and the Brooklyn Bridge in New York are both suspension bridges, and the same basic design patterns were used for both.

The difference between reusable "Universal Data Models" and "Universal Patterns for Data Modeling" is that Universal Data Models apply to very common, specific models, such as a party model, order model, or telecommunications model, whereas the Universal Patterns can be used to extend and develop just about any type of data model. This does not imply the Universal Patterns and Universal Data Models are incompatible, in fact, most of the data models in the first two volumes of the *Data Model Resource Book* series used Universal Patterns in their construction. We believe that Universal Patterns can enhance the building of any data model.

In general, a pattern is something intended as a guide for making something else. A pattern in data modeling can be described as a template that can serve as a guide for developing data models. For example, the status pattern in Figure 1 provides a guide or template for modeling the statuses for any type of entity. Thus, the status pattern may apply to the status of a PARTY, PRODUCT, ORDER, INVOICE, or any other entity that has various states. A PARTY may have states or statuses of "Active," "Inactive," and so on, and a PRODUCT may have statuses of "Introduced," "Support discontinued," and so on. Both of these entities could use the same "pattern" to model their states.



Notes about this data modeling notation.

The dotted line indicates optionality (as opposed to mandatory) for each side of relationship.

A crow's feet (the three prongs at the end of the relationship) that there are many occurrences of instances of an entity. For example in the center of Figure 1 "each ENTITY *may be* **in the state of** one or more ENTITY STATUS".

A "*" in front of an attribute indicates an attribute is mandatory, an "o" indicates an attribute is optional.

(PK) indicates that an attribute is a primary key.
(FK) indicates that an attributes is a foreign key.
(UID) indicates that an attribute is part of a unique key.

"............................" indicates that there may be more attributes in this entity that are not shown in the diagram.

Figure 1, Status Pattern and examples of using the Status Pattern

The Universal Patterns for Data Modeling represent effective practices and alternatives for modeling very common types of data models. Another example of a Universal Pattern is the underlying data model showing how a PARTY is related to other entities. For example, in Figure 2 we see how PARTY is related an INVOICE is very similar to how PARTY(s) are related to SHIPMENT(s), which is also very similar to how PARTY(s) are related to PROJECT(s), or any other entities. Parties (people or organizations) may have certain roles within the context of a particular transaction or with regards to another entity, and there are very common ways to model this type of data requirement. We call this particular example the Contextual Role Pattern.

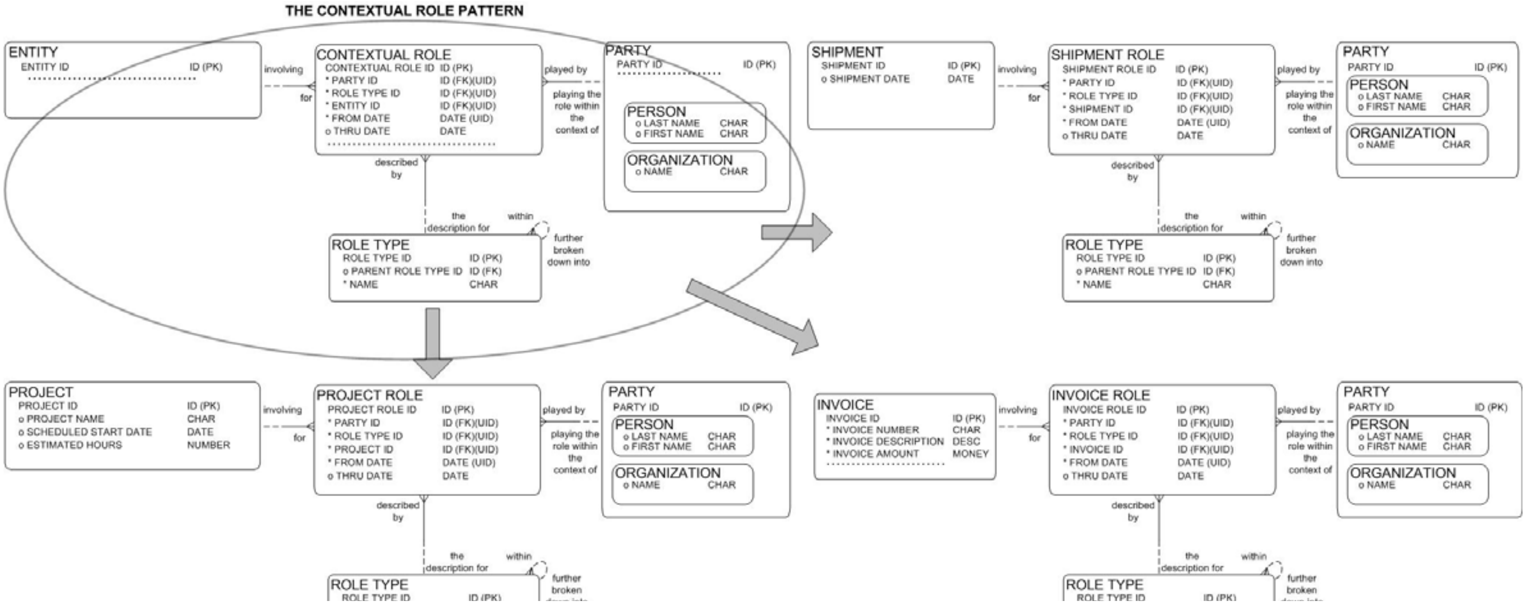


Figure 2, Contextual Roles Pattern and examples of using the Contextual Roles Pattern

In Figure 1 and Figure 2 we show two different types of Universal Patterns (status and contextual role), but these are not the only “flavors” for these patterns. Much like there are many different bridges (cantilever, suspension, pontoon) or many different designs for a chair or a table (art deco, shaker and so on) there are different styles of modeling the same Universal Pattern. We refer to these as different levels of generalization. For example, a level 1 pattern is the most specific pattern, where as a level 3 pattern is more generalized. Each of the patterns evolves from a specific pattern to a more and more generalized pattern. For example in Figure 3, we see a very specific pattern (level 1) that supports hierarchies (and aggregations) and in Figure 4 we see a generalized version of the same pattern (level 3).

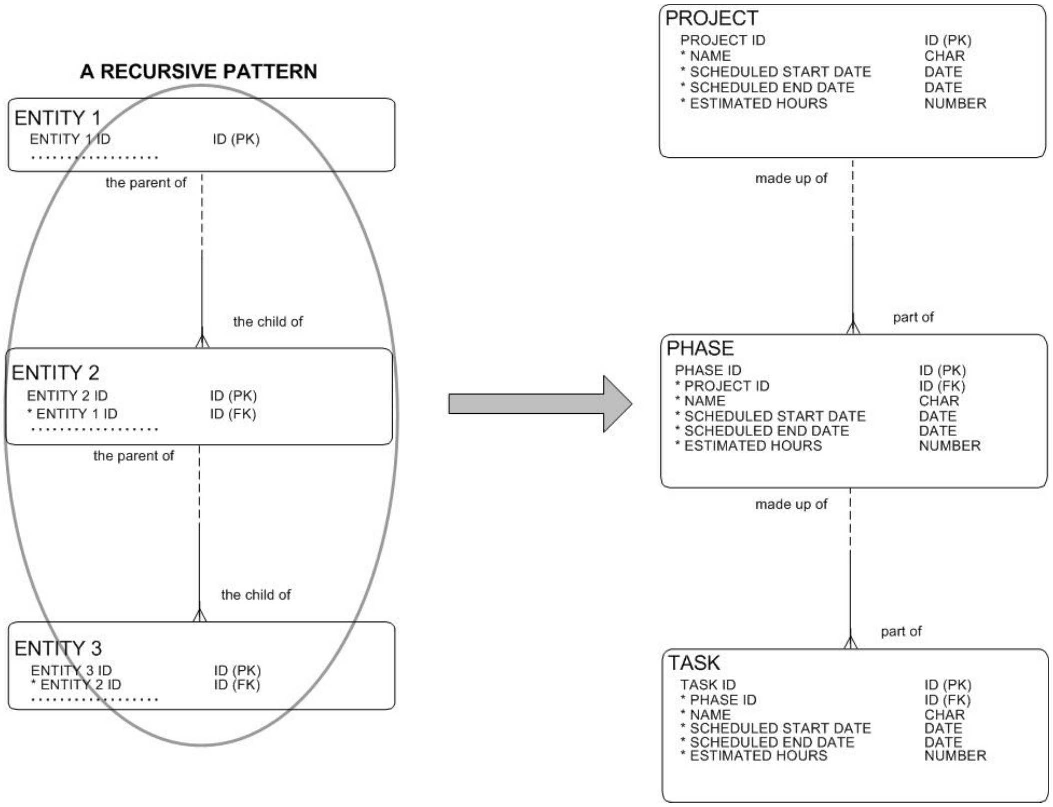


Figure 3, a specific Hierarchy/Aggregation Pattern

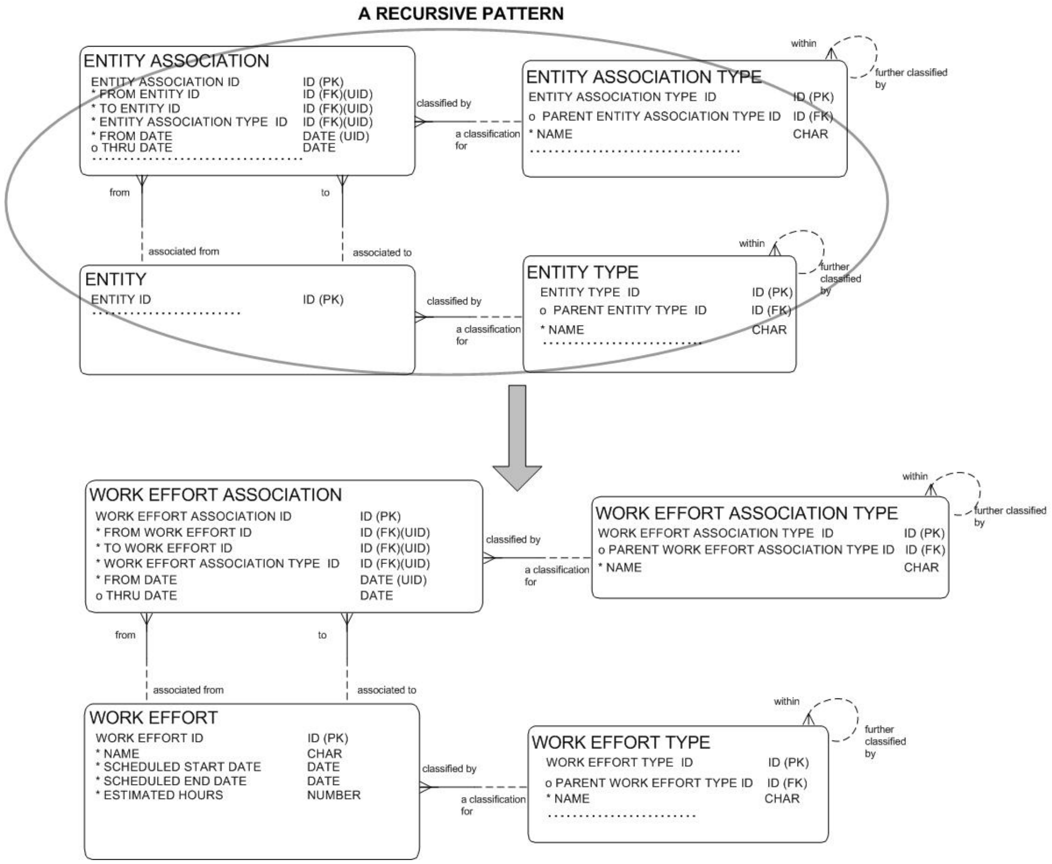


Figure 4, a Generalized Hierarchy/Aggregation Pattern

Initially, in Figure 3, Hierarchies and Aggregations are handled in a specific manner by modeling them as specific entities. We characterize this as a very specific level 1 pattern. Subsequently, each of the patterns becomes more and more flexible in its approach by using more generalized data model constructs to model this same type of information. Thus, you can create level 2 patterns, which generalizes the pattern, then the level 3 pattern, which is a more generalized version, and a level 4 pattern which is even more generalized. The level and style of pattern that you may choose to use depends on the needs of the enterprise being modeled and the circumstances involving the modeling task.

NOTE

NOTE

We have used the word *generalization* instead of using the more common term *abstraction*. Many data professionals believe that *abstraction* implies a loss of detail. For example, a map of a roadway is an abstraction because it limits details to a certain level in order to focus attention on roads. *Generalization* implies transforming very specific data model structures to more generalized concepts, in order to more flexibly support data requirements. *Generalization* provides this flexibility by using a less

> specific data structure and accommodates current and future requirements via adding, changing, or deleting data instances. Another reason for using the term *generalization* is that the object-oriented community uses the term *abstraction* in a different way that has a different meaning.
>
> Also, *generalization* should not be confused with *normalization*. They are completely separate concepts. Generalization has to do with using more flexible data model constructs, whereas normalization has to do with eliminating data redundancy by relating data to "the key, whole key, and nothing but the key."

How can you answer the question whether to use a specific pattern or a generalized pattern? You can first ask the question, "What is the purpose of a data model?" We believe that there are two key purposes to a data model:

1. To illustrate and communicate information requirements.
2. To provide a sound foundation for a database design.

These purposes can be at odds with each other. If the purpose is to illustrate and communicate information requirements, the modeler will most probably develop a more specific model showing the specific needs of the business representative. For instance, in order to define the information requirements for a PROJECT, PHASE and TASK hierarchy, the modeler may show figure 3. Accordingly, this would be considered a specific style of modeling and would be a level 1 pattern. However, if the purpose of the data model was to provide a sound foundation of database design for any WORK EFFORT relationships then the level 3 pattern would be more suitable, as seen in Figure 4.

What we have found based upon decades of data modeling experience is that the same types of patterns continually occur in data modeling efforts. In our new book, Volume 3 of *The Data Model Resource Book: Universal Patterns for Data Modeling*, we have chosen what we think are the most common, "universal" patterns in data modeling. We have found that a great majority of the data modeling in most organizations have to do with roles (what parties do and how parties are involved), hierarchies and recursions (the organization of similar data), types and categories (the classification of data), statuses (the state of data), contact mechanisms (how to get in touch), and business rules (how things should work).

Why do we refer to these patterns as universal? The word *universal* is defined in Webster's dictionary as "applying to a great variety of uses: comprehending, affecting or extending to the whole." Thus Universal Patterns for Data Modeling are reusable guides that provide a data modeling template for very prevalent or "universal" themes that occur in data modeling. The intent of these patterns is that they can "apply to a great variety of uses" and that they can be used by a great number of organizations to save time and effort while offering holistic (that is, universal) perspectives.

Universal patterns for data modeling can be used to build upon common models in a consistent fashion, with the confidence of knowing that the patterns are true and tested common constructs that work in real life. Many of our clients have used these patterns in many different ways, for example:

- To provide a standard that IT professionals can adhere to when modeling data. This has helped them keep a consistent style for data models and subsequent data structures across different databases in their organization.
- As a standard toolkit that data professionals can turn to when building/extending their data models. The patterns cover many of the standard problems that data modelers need to address. Why solve the problem again when the patterns already give you different flavors (levels of generalization) of the solutions, thus providing effective alternatives with their pros and cons?
- As a standard that can be used as the basis for common database structures that allows developers or programmers to create standard interfaces to and from these common structures that are based on the patterns. This means that programmers can "program to the interface" and have less concern about dealing with many different underlying data semantics and data structures.
- As a useful tool for clients when they buy software or other standard data models. The patterns can set the data requirements that a vendor data model or database must rise to regarding very common data needs. For example, the patterns can specify that a solution needs to support very flexible classification schemes that allow new types of classifications without changing the data model or data structure; does the product you are looking at accommodate this type of flexibility regarding maintaining classifications? Does it use flexible patterns? If it does not, how does it provide the appropriate level of flexibility that you need?
- As an objective source against which an enterprise can evaluate and check its data models from its previous systems development efforts so it can evaluate alternative options.
- As training materials for their data professionals and IT staff in general. The patterns cover a broad range of different structures at different levels of generalization. The patterns are explained in detail with examples that can guide data modelers and other IT professionals in their use.

Many of our clients have used these patterns successfully to save time and increase the quality for a great variety of data modeling efforts, ranging from creating a data model for a prototype, through developing an enterprise-wide data model used to standardize their models worldwide.

Published in the B-eye Intelligence newsletter