



# Domain Modeling



**Dennis Pierson**

Senior Information Systems Architect & Head of Semantics  
Mphasis



**MPHASIS**  
*an HP company*



## Contents

<b>1. Introduction</b> .....	<b>4</b>
<b>2. Embrace Diversity</b> .....	<b>4</b>
<b>3. Rationalize Data</b> .....	<b>5</b>
<b>4. Align Architecture with Business Concepts and Processes</b> .....	<b>5</b>
<b>5. Domain Modeling</b> .....	<b>6</b>
<b>6. Building a Domain Model, an Abbreviated View</b> .....	<b>6</b>

# 1. Introduction

Data modeling and engineering are highly technical pursuits that don't concern most business systems users. The modelers and engineers who design and implement database schemas normally do so according to individual application requirements. The resulting proliferation of inconsistent data models has led to a realization that important entity models should be standardized across applications as canonical entity models rationalized across the business. This concept lies at the heart of master data strategies. Many of these schema unification projects break down when canonical models are applied across business boundaries because they cannot embrace the diversity of business semantics as they are represented in different parts of an organization. There really is no single view of a customer.

We are often asked, even expected, to bring reference database schemas to our engagements. Our experience with standardized models has shown us that such an approach is inadequate. It's just never that simple. These models are always based on assumptions about the processes and applications that they support and the notion that a single model can serve multiple different points of view.

A canonical reference model applied for different applications and business groups must be reviewed carefully in detail in every context where it will be applied. The extensive adaptations and changes required to fit the model to real-world situations defy the idea that this approach is practical; and the result is usually not pretty.

We don't believe there is a single canonical way to model different real-world points of view in practice using relational database schemas. The relational model is simply too constrained semantically to accommodate the differences among applications and users. Failure to reach agreement or even consensus among disparate stakeholders poses a high risk to the success of a project. We take a higher-level and more fundamental approach: embrace diversity with rich semantics using Domain Modeling.

## 2. Embrace Diversity

Take the customer entity for example. A single customer looks a lot different to a sales system than it does to a billing or credit review application. Different groups use different language; each has its own shorthand jargon, some with deep roots in the business culture. In some cases a term referring to an entity attribute or a process concept may be applied in a completely different way in different semantic subdomains. Trying to achieve term agreement among business users at the technical level of a database schema doesn't make sense.

What does make sense is the development of a controlled vocabulary to which the different sets of terms can be related in a purely semantic model, thereby correlating the different conceptual subdomains in a semantically rich domain model. This provides a clear and easily understood elaboration of the business vocabulary and the taxonomies of those terms, and it exposes the relationships among them that describe the business model.

Consider also the notion that a customer entity may be a complex object with distinct type variations, each with a distinct attribute set. In a single-entity model, the superset of attributes will be modeled, and the entity assigned a type indicator. In fact, this oversimplifies to the point of unnecessary confusion. These customers really are different types, and should be modeled as class objects. Domain models embrace this kind of rich diversity without collapsing the semantics.

### 3. Rationalize Data

A data governance issue common in data migration and rationalization projects, including MDM, is the gap between the conceptual view of data and the actual, empirical use of the data. Data quality analysis will discover gaps and errors in data, but before launching a project to fill and correct, we use process mapping to discover the actual production value of data as a standard step in data quality analysis.

In one instance, we discovered that the age value of customers was grossly under-populated. Process analysis revealed, however, that this element was relevant only to credit applications, and wasn't required elsewhere. Marketing demographics didn't require age data to be populated.

So, let's suppose that we can establish a semantic baseline for understanding, modeling, and managing our data. That's the topic of this paper: domain modeling as an architecture practice to establish and maintain alignment between business, data governance, and the automated processes we all depend on throughout and beyond application and system lifecycles.

### 4. Align Architecture with Business Concepts and Processes

A primary dimension of the practice of Enterprise Architecture is strategic alignment. For an Architect to create an adaptive and enduring architectural model, it is important to be aware of enterprise business strategy. This principle applies at any granularity of automation as contextual alignment.

Every application, every modernization, every IT project takes place in some business context. If it is implemented without contextual alignment, much of the knowledge gathered to define the application will not be captured, and it will not adapt well to changes in business processes.

Fundamental to any automation project is requirements capture. Somewhere, sometime later, usually after elaboration and some engineering design, a logical data model emerges. To a business person, this is a relatively arcane IT artifact. Validating this against requirements requires reverse engineering the data model, even if traceability has been documented. The process is difficult, tedious, and not reliable because the inherent domain model has already been semantically reduced and the context disposed.

Well established protocols for defining and developing software, such as SDLC, concern specific sets of specifications that are focused solely on the instant application. The static handoff models that develop to a specification are usually stripped of business context during the specification process. Even in RAD and agile processes where business experts collaborate with engineers to craft specifications during development, the business context is not normally captured, and application design doesn't align beyond the immediate application specification. The subject matter experts move on.

In our architecture practice, we start with the domain model. It is fundamental to the requirements gathering process because it establishes a transparent foundational model expressed graphically in business language that can be validated by business experts. Changes in requirements made to the domain model ripple through the rest of the process naturally, maintaining and ensuring alignment of technical specifications to business requirements.

The business domain model provides a master data model from which the logical and physical data models can be derived and validated. It manifests a bedrock frame of reference for the rest of the SDLC process through development, testing and implementation, and beyond. The Domain Model continues to serve as fundamental

governing application documentation that informs, and aligns throughout an application's lifecycle - modifications, enhancements, upgrades, migrations, and replacements, and then beyond the lifespan of individual applications and business processes. A well-formed domain model captures the enclosing context and the domain relationships with the business model, and persists as a solid and slowly changing basis for agile business process execution.

Domain models offer such benefits as

- Agreement among stakeholders about domain vocabulary and business semantics
- Alignment of business intention with IT delivery
- Rationalized data and governance models

## 5. Domain Modeling

A domain model is a model of the essential knowledge of a business. It captures the semantics of the domain – the meaning of the entities and the relationships (not relations) among them. A domain model includes two primary component models that describe structure and behavior respectively, and the mappings between them in clear graphical notations that can be understood and validated by both technical and non-technical people.

The structural component is a semantically rich data model described in business terms. This model forms an enduring basis for agreement between business stakeholders and technical people about the business context that surrounds and determines the implementation and use of data and the processes that handle the data.

The behavioral component describes the processes that use the data objects in the structural model. We also model state transitions to tease out object lifecycles and stakeholder use cases that align with state transition triggers. These results are used to enrich the domain model and to understand how data is actually used. As many of us are aware, data is messy stuff. Process models reveal important information about where data is sourced and processed, where it is required, by whom, and how.

As in a business with divisions, products, markets, and so on, the knowledge for each will describe an individual domain, or subdomain. There is no single a-priori rule for bounding a domain, except as in object modeling; the boundaries should be 'crisp', and apply succinctly to the project at hand. The expertise of the stakeholders and their subject matter experts usually suggests the limits of a domain. One of the best features of domain models is that they are flexible - they can be merged and partitioned as required to adapt to evolving business model changes.

## 6. Building a Domain Model, an Abbreviated View

### 1. The Static View: Model the business domain

- Capture the facts of the business domain in a concept model. It is a basic tenet of semantics that context matters. Context is metadata, in fact - Zen Buddhism avows that nothing has meaning without context. Application boundaries may shift over time. Functionality may be partitioned or aggregated in subsequent generations. Context does matter.
- Model these facts in Object - Role Model (ORM)<sup>1</sup> or a similar concept notation (not ER). A good rule of thumb is that nouns are entities, verbs are roles, adjectives are attributes, and prepositional phrases are qualifiers that suggest n-ary relationships. This elaborates the relationships, or roles, between the entities. ORM was developed as a graphical notation for the design of relational database schemas. We use it to expose relationships among the objects in a domain to reach clear agreements among business experts, stakeholders, and between them and technical people.

<sup>1</sup><http://www.orm.net/>

## ORM Sample...

- f1 The *Academic* with empNr 715 has *EmpName* 'Adams A'.
- f1 The *Academic* with empNr 715 works for the *Dept* named 'Computer Science'.
- f1 The *Academic* with empNr 715 occupies the *Room* with roomNr '69-301'.
- f1 The *Academic* with empNr 715 uses the *Extension* with extNr '2345'.
- f1 The *Academic* with empNr 715 provides the *AccessLevel* with code 'LOC'.
- f1 The *Academic* with empNr 715 is contracted till the *Date* with mdy-code '01/31/95'.

### A set of facts

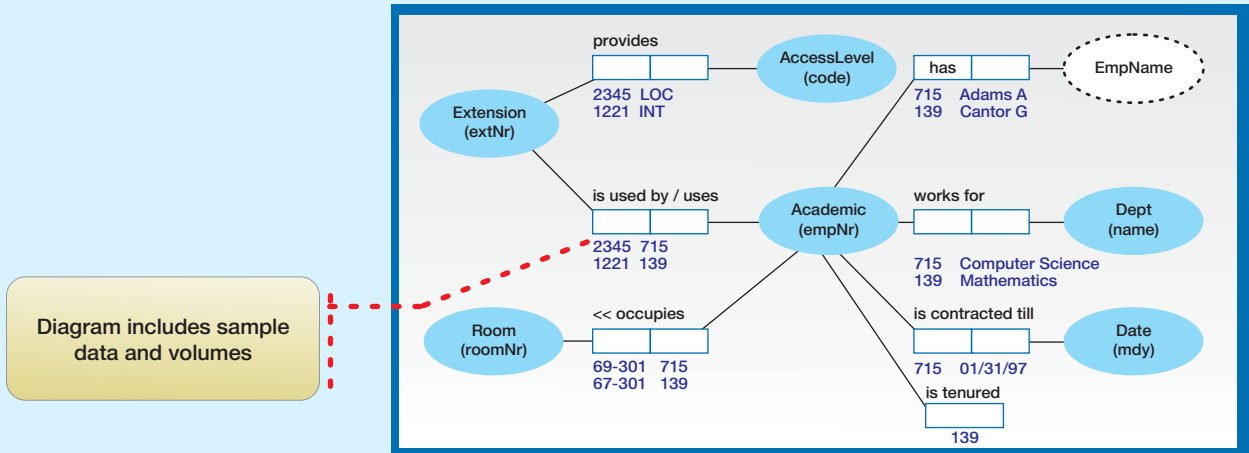
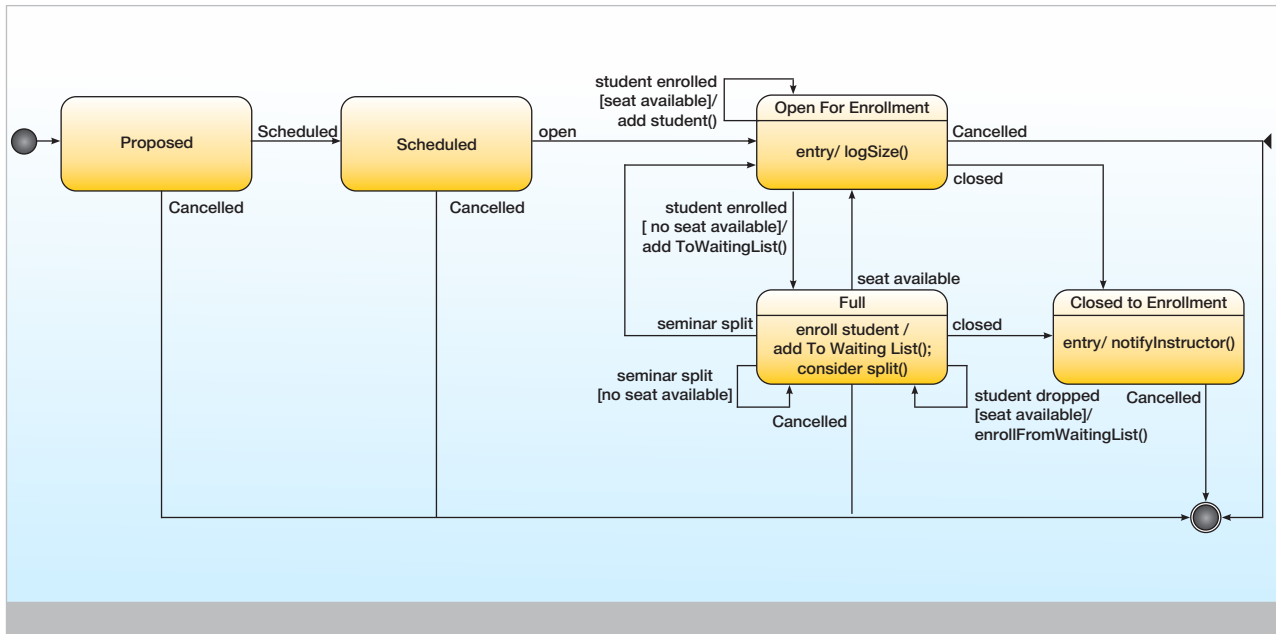


Figure From "Object-Role Modeling: an overview"<sup>2</sup>

## 2. State View: Model state transitions for key entities

- Show key states and state transitions
- Capture state transition triggers as annotations



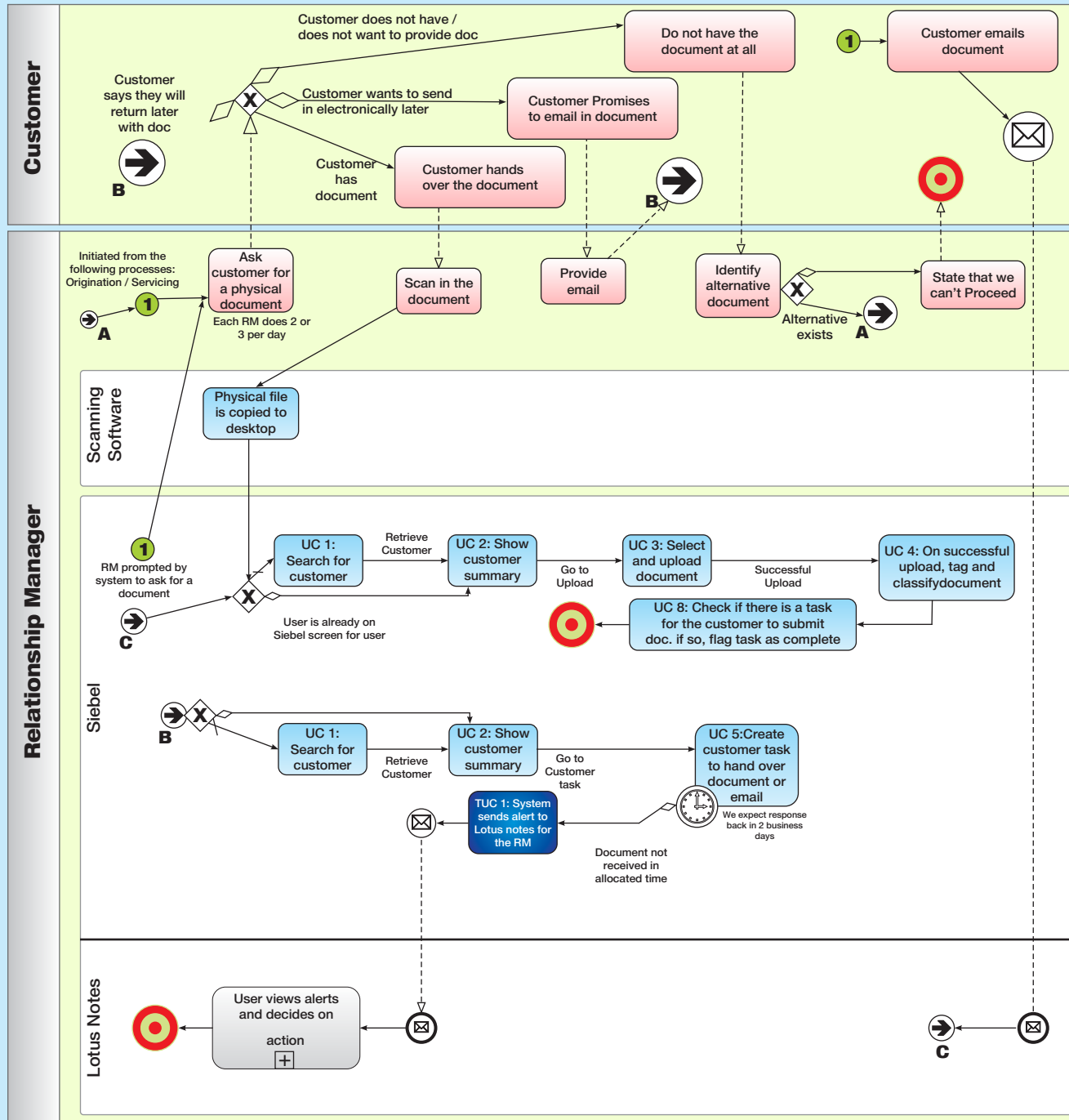
<sup>2</sup>"Object Role Modeling, an overview", Terry Halpin, 2002 - then with Microsoft

### 3. Participation view of stakeholder use cases:

- Identify actors and use cases
- All state transition triggers must have corresponding use cases

### 4. Process Views: model “as-is” and “to-be” business processes in BPMN<sup>3</sup>

- Process models show how data is processed and where the human-human, human-system, and system-system interactions take place. This exposes how data is used, and provides an empirical valuation for the elements.
- Detail human interactions by role, external systems and their interfaces, messages, and repositories. Timing and synchronization constraints including parallelisms will be exposed along with the major system components in their respective functional layers.



<sup>3</sup><http://www.bpmn.org/>

We've seen great interest in a semantic approach to business-model-driven IT projects and in domain models generally, as more and more people understand the need and importance of understanding and communicating their knowledge domains. Domain modeling can be an arduous and long-winded affair, requiring collaboration by numerous people with different points of view, but it can begin with a simple statement of facts, and proceed incrementally. The model doesn't have to be richer than its intended use. It can be built out as needed and as resources are available without breaking any dependencies that have already been established.

Industry efforts to develop standard models are emerging in several verticals. The Enterprise Data Management Council has developed a very rich and detailed model for Financial Services driven primarily by requirements for better regulation. This model is publically available at their working site<sup>4</sup>.

<sup>4</sup><http://www.hypercube.co.uk/edmcouncil/>

## About the Author



### **Dennis Pierson**

Senior Information Systems Architect in the Financial Services Architecture Group, and Head of our Semantics Practice

With a career spanning 30 years in the software industry, Dennis has worked as a consultant and in product development in Financial Services, Pharmaceuticals, Insurance, Defense, and Manufacturing. He heads Mphasis' Semantics Lab, using Domain Modeling as a core architecture practice, and bringing Semantic Web knowledge-based solutions to enterprise IT.





# ABOUTMPHASIS.

Mphasis is a \$1 billion global service provider, delivering technology based solutions to clients across the world. With over 41,000 people, Mphasis services clients in Banking and Capital Markets, Insurance, Manufacturing, Communications, Media & Entertainment, Healthcare & Life Sciences, Transportation & Logistics, Retail & Consumer Packaged Goods, Energy & Utilities, and Governments around the world. Our competency lies in our ability to offer integrated service offerings in Applications, Infrastructure Services, and Business Process Outsourcing. To know more about Mphasis, log on to [www.mphasis.com](http://www.mphasis.com)

For more information, contact: [sales@mphasis.com](mailto:sales@mphasis.com).

USA: 460 Park Avenue South, Suite #1101, New York, NY 10016, USA  
Tel.: +1 212 686 6655, Fax: +1 212 686 2422

UK: 88 Wood Street, London EC2V 7RS, UK  
Tel.: +44 20 85281000, Fax: +44 20 85281001

AUSTRALIA: 9 Norberry Terrace, 177-199 Pacific Hwy, North Sydney, 2060, Australia  
Tel.: +61 2 99542222, Fax: +61 2 99558112

INDIA: Bagmane Technology Park, Byrasandra Village, C.V. Raman Nagar, Bangalore 560 093, India  
Tel.: +91 80 4004 0404, Fax: +91 80 4004 9999



1011



MPHASIS  
an HP company