White Paper

embarcadero

Considerations: Mastering Data Modeling for Master Data Domains

David Loshin

President of Knowledge Integrity, Inc.

June 2010

Americas Headquarters

100 California Street, 12th Floor
San Francisco, California 94111

EMEA Headquarters

York House
18 York Road
Maidenhead, Berkshire
SL6 1SF, United Kingdom

Asia-Pacific Headquarters

L7. 313 La Trobe Street
Melbourne VIC 3000
Australia

Introduction: Challenges of the "Master Directory"

Increasing dependence on enterprise-class applications has created a demand for centralizing organizational data in their support. Imperatives such as enterprise resource planning (ERP), data warehousing for business intelligence, and customer relationship management (CRM) rely on data integration programs such as customer data integration (CDI) and master data management (MDM), which essentially is a collection of data management techniques used to facilitate the definition and observance of policies, procedures, and infrastructure to support the capture, integration, and sharing of a trustworthy set of unified views of master data concepts.

These master data concepts, such as *customer*, *product*, or *employee*, are those core business objects that are used in the different applications across the organization, along with their associated metadata, attributes, definitions, roles, connections, and taxonomies. Master data objects are those "things" that we care about – the things that are logged in our transaction systems, measured and reported on in our reporting systems, and analyzed in our analytical systems.

Although some of the objectives of master data integration include improved data quality and operational efficiency, the development of master data indexes, registries, and hubs is often complicated by challenges inherent in the organic manner in which the *de facto* enterprise application infrastructure evolved. These challenges, which are magnified when developing a multi-domain master environment that incorporates hubs for each of a number of master data concepts, are a byproduct of diminished oversight over shared organizational information and data modeling.

In this paper, we explore some of the root causes that have influenced an organization's development of a variety of data models, how that organic development has introduced potential inconsistency in structure and semantics, and how those inconsistencies complicate master data integration. A particular concern is that the desire for a master data environment managing multiple data concepts allows duplication to occur. But by applying a governed approach using universal models, the data engineers can reduce the risk of duplication and inconsistency while improving the quality of the process and the results of master data integration.

Root Causes Drive the Need for Directed Modeling

The root causes are not issues while data sets remain tied to their corresponding applications, but when attempting to consolidate data into shared resources these emerge as significant barriers to success. Some prime examples we explore in this paper include:

- Conflicting structures and semantics, in which similar concepts are represented slightly differently, and may have variations in their underlying meanings;
- The expectation of singularity of master entities, suggesting that any real-world object is represented in one and only one master record;
- The need to create horizontal views across master data domains to provide visibility across business applications; and
- Issues with aligning vendor master model semantics with the existing variety of internal models.

Conflicting Structure and Semantics

Historical technology development trends have influenced the current state of an organization's information architecture. The 1980's saw the rampant deployment of low-cost workgroup computing resources supporting business applications for individual groups within an organization. A byproduct of this trend was the development of decentralized business processes and their associated applications. Correspondingly, this led to diversity in representations across the underlying data models.

Because data modelers were directly supporting workgroup computing requirements, they concentrated on developing data elements, entity tables, relational structures, and, correspondingly the semantics specific to their own applications. Although this decentralized approach was nicely suited to meet immediate operational and transactional business needs, the organic evolution of the underlying data models led to information model entropy, incremental data model inconsistency, and variations in structure and semantics. Some common examples include:

- Variations in data element types and sizes such as numeric vs. alphanumeric for identifier values;
- Overloaded data attribute use, in which the same attribute is used for multiple purposes, such as storing email addresses in a FAX number attribute;
- Different reference data domains used for the same conceptual domain;
- Different table structure and corresponding variances in relational structure.

The variations are not that important as long as there is no need to share data. But as soon as there is a desire for a unified view of master business concepts across vertical silos, these structural and semantic variations can become a true roadblock.

Singularity of the Master Entity

There is an inherent inequity engineered into the concept of multi-domain master data management. From one perspective, the concept of a "master" data repository implies that there is one and only one instance of each entity within the organization's data environment. But from the other perspective, the concept of multiple master data domains suggests a master data organization centered on representing specific data entity types such as *customer*, *product*, *employee, part, vendor, member,* etc.

While the multi-domain approach does support a relational view of interactions among different master entity types, the siloed data organization conflicts with the idea that the same real-world thing might exist more than once in different business contexts. For example, it is not unusual in many companies to find employees that are also customers, and there are certainly companies that manufacture and sell the same parts they themselves use in building other products.

In essence, by aligning master domains with the roles the underlying entities play, we open the possibility of duplicating common attributes associated with the same entities stored in different data repositories. Not only does this violate the goal of uniquely identifying each entity, it can lead to inconsistency of common attribute values across the different domains.

The Horizontal View Across Domains

Typically, each business application relies on its own data silo to capture information about the entities that participate in the business processes, providing a vertical view of the data. However, a common driver for master data management is transparency across a collection of data silos. When data consumers express a desire for this horizontal "360-degree view" of the customer (or employee, vendor, etc.), they expect visibility of attribute values and interaction histories collected from across business application boundaries. For example, a sales agent may want to see a complete inventory of all products each customer has purchased, even if those purchases were made through different channels.

When individual master domains are maintained as separate master repositories, there remains a distinction between each entity in the context of the domain (e.g., customer or employee). Without a process for establishing a connection between the corresponding underlying entity instances, the ability for any application to materialize that 360-degree view is impaired. This horizontal view is only available when there is explicit knowledge that the same party acts in different roles, even in different business contexts.

Vendor-Driven or Model-Driven?

An organization that has opted to implement a master data program is likely to engage vendors and select MDM tools to support the program. And since many MDM tools and technologies are intended to deliver a full solution, these products are likely to incorporate vendor-provided data models for the master data concepts deemed critical (by the vendor) to an enterprise. This "one-size fits all" approach is perhaps a bit presumptuous, especially considering how subtleties and variance in existing data model structure and semantics pose integration challenges. Adding in yet another (external) model may actually add to the confusion, not resolve it.

As a result, deeper question emerges: should master repository structures be driven by externally-defined vendor models? In some situations, that approach may be sufficient, but more sophisticated organizations require underlying master models that can accommodate the existing data, business applications while providing a streamlined path to meet future data and business application needs. This model-driven approach means considering a combination of current and future enterprise data requirements, and then developing multi-domain master entity relationship models that capture the critical core concepts, roles, interactions ,and relationships as a way to finesse the issues we have explore so far.

Master Data Concepts vs. Master Data Models

It is valuable to recall that there is a practical difference between conceptual master data items and the ways those items are implemented within a logical data model. Despite the intent of multi-domain environments to capture information about a siloed collection of data concepts such as customer or product, enabling visibility into the relationship between the organization and the different data concepts across the enterprise requires a pragmatic approach that segregates the underlying entity (such as a specific individual) from the different roles that entity can play (such as customer, vendor, or employee.

What is a Customer? And What About an Employee?

Our expectation for master data concepts is that they center on the collection of unique entities belonging to a particular class of objects. This is the reason that organizations focus on building a "product" or "customer" master resource. Yet the quandary remains: does the master data concept prescribe the model, even when it leads to a violation of the objective of unique representation for any real world entity?

This can be demonstrated using the example of an individual who is both an employee and a customer. Storing the core identifying and demographic attributes (such as name or birth date) associated with the individual within both the customer and the employee master repositories means data duplication. In other words, driving the creation of the master

repositories based on the proposed master data concept exposes this potential replication. Therefore, when the same entity exists in multiple domains, there must be some approach to modeling that does not violate the uniqueness objective.

CUSTOMERS? OR ENTITIES AND ROLES?

The alternative is to engineer master data domains so that they can observe the uniqueness requirement while allowing for the differentiation of representation by domain. Recognizing that there are core entities that can play different parts depending on the circumstance, a universal modeling[1] approach allows for the definition of specific entities that can be cast into a variety of specific roles depending on the business application.

To continue our earlier example, instead of replicating data about that individual employee who is also a customer, we can create a "party" entity model that contains the identifying and demographic attributes that we'd prefer to store once only. In turn, we can map those parties into specific roles (such as "customer" or "employee," etc.) and associate role-specific data attributes within that augmented model. As another example, we can define a "component" entity model capturing the characteristics engineering that can act in the role of "product" when it is being sold or act in the role of "part" when it is to be used as part of internal design and manufacturing processes.

Multiple master domains are then managed by mapping unique entities by role across the application framework. Common master data services can be layered in relation to the entity definition hierarchy definition, like the example for new customer creation suggested in the process flow shown in Figure 1.

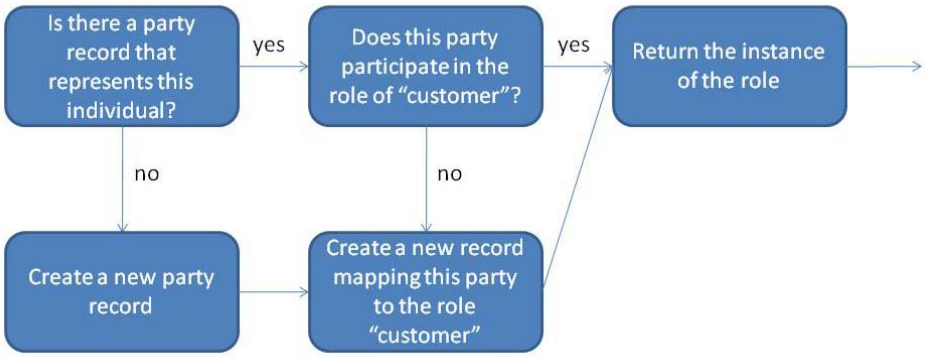

Figure 1: Layering new customer creation on top of party record creation.

[1] For more information on universal modeling, consult the excellent "Data Model Resource Book" series by Len Silverston.

Exploiting Relations, Relationships and Cross-Domain Visibility

This approach does more than allow you to observe the uniqueness requirement for master data. Adopting the universal modeling approach enables the creation of model that will accommodate the materialization of a complete ("360-degree") view of any specific instance of any of the defined entities. Instead of maintaining a view along each domain that slices along a single master data concept, one can consider how a single entity participates in various roles with other entities across multiple business contexts to provide cross-domain visibility.

Better yet, one can extend this modeling approach to map different relationships within and across the different master domains, such as household and family relationships for individuals, or product categories. This provides the ability to better trust answers to questions analyzing the relationship across domains, such as:

- How effective were our engineers in designing products using our own components?
- How much of a company's business is associated with its own employees?
- How influential are our employees in recommending business to their relatives and friends?

In turn, these models allow for more believable reporting and analytics.

First Things First: Requirements, Use Cases, then Models

The universal modeling approach helps to alleviate some of the hazards of inconsistency and confused semantics introduced through haphazard master data consolidation. However, this approach to master data domain modeling cannot be performed in a vacuum – it requires some prerequisite work to be performed, focused on identifying master data concepts, clarifying their corresponding class hierarchies, and accurately soliciting business consumer data requirements prior to developing the model. We can articulate these five preparatory steps prior to developing the models:

- Catalog – In order to review the class hierarchies ultimately to be represented as master data domains, it is important to identify the master data concepts that are used across the enterprise. During this step, the critical business processes are analyzed to note the data concepts that are candidates as master data. Generally, those data concepts that operate within more than one business application and have attributes in common across those applications are candidates.
- Differentiate – During this step, the data analysts review the characteristics of the candidate master data concepts to differentiate the core entity from the role that each

entity plays in the different business contexts. This allows the analyst to distill out the identifying attributes that uniquely describe the core entity as well as determine which attributes are associated with the role each entity plays.

- Map – While the previous step is used to unravel the class hierarchies, this step is used to understand the relationships among the different entities within and across contexts. These mappings describe the business connectivity among core entities, and can be articulated as simple assertions using nouns and verbs (" *customer* ***buys*** *product," "employee* ***is contacted using*** *telephone number").*
- Solicit –The business data consumers are engaged through directed interviews to solicit information requirements relating to operational and analytical business application usage of the master data concepts in the context of those relationships noted during the **map** step.
- Document – Finally, the data analysts must clearly articulate the class hierarchies, relationships, and business consumer requirements and directives to guide data modeling and master system development .

EMPLOYING THE RIGHT TOOLS

Although organizations increasingly rely on MDM to support enterprise imperatives such as ERP and CRM, many programs stall due to structural, semantic, and quality issues stemming from organic model development. We have suggested that leveraging a universal modeling approach that segregates entity from role allows a hierarchical representation that can provide the horizontal visibility, delivering on the promise of multi-domain master data management. This alternative, model-based approach demands that the enterprise information modelers arm themselves with the right skills and tools to develop and manage these master data models including data assessment, metadata management, and especially data modeling tools.

BOTTOM LINE

You're looking at a long and complex MU conformance process filled with many complicated projects and very little time to get your systems up to meaningful use standards. If you start with good planning, move to analyze your workflows and functional requirements, craft an effective change management strategy, and put in good tools and techniques for an effective implementation approach you'll be able to meet the requirements put into place by the government and get better systems for your customers.

About the Author

David Loshin is the President of Knowledge Integrity, Inc., a consulting and development company focusing on customized information management solutions including information quality solutions consulting, information quality training and business rules solutions. Loshin is the author of *Master Data Management*, *Enterprise Knowledge Management – The Data Quality Approach* and *Business Intelligence – The Savvy Manager's Guide. He has an expert channel on BeyeNetwork where he blogs on* information quality and is a frequent speaker on maximizing the value of information.

Embarcadero Technologies, Inc. is the leading provider of software tools that empower application developers and data management professionals to design, build, and run applications and databases more efficiently in heterogeneous IT environments. Over 90 of the Fortune 100 and an active community of more than three million users worldwide rely on Embarcadero's award-winning products to optimize costs, streamline compliance, and accelerate development and innovation. Founded in 1993, Embarcadero is headquartered in San Francisco with offices located around the world. Embarcadero is online at www.embarcadero.com.