

# A LEAN APPROACH TO MASTER DATA MANAGEMENT

---

BY [DUFF BAILEY](#)

FROM THE CUTTER IT JOURNAL - 26 MARCH 2008

Enterprise IT leaders who seek customer value from a master data management (MDM) project [1] can find themselves in a Catch 22. They can't put all development on hold while they wait for a full enterprise data model to be developed and approved, yet they know anything that is developed in the interim will be subject to a costly, lengthy, and, quite possibly, ugly remediation when the new standard is available.

Agile data management (ADM) [2] offers an evolutionary approach to MDM that focuses on effectively solving the problem at hand and evolving toward an overall data model. While there will be times when the model must change and refactoring is needed, agilists posit that the time and effort to do this will be less than the previously mentioned cost of refactoring once the BMUF (big model up front) is finally ready. They also make a valid point that however good the big model is in theory, its suitability is never tested until it's implemented -- and that inevitably leads to refactoring anyway.

Even if you accept that premise, however, the agile MDM approach still poses problems in an enterprise setting. Separate teams will align with separate stakeholders -- and chances are they will evolve in contradictory directions that reflect the concrete aspects of their stakeholders' current business challenges [3]. The end result can well be a patchwork of systems that don't play well; not unlike all one's other legacy systems.

In truth, both agile and traditional approaches to enterprise data management suffer from the same flaw: they anchor their models off of the current state of the business. Hence, the models will break when the business changes -- which successful business do, radically and quickly. The financial giant, JPMorgan Chase, started as the Manhattan Water Company -- while Apple has morphed from computers to entertainment and media.

The master data model needs to be anchored off of an abstract picture of the enterprise that can apply universally to any business model. In this view, a business is an organization that creates an infrastructure and pays people to transform and transport, to other locations, products and services purchased from some people and organizations so that they can be sold to others. The key objects in this model are organization, infrastructure, people, locations, products, and services. These are your core entities and they will appear repeatedly in business transactions, which are interactions between specific instances of these elements.

A lean MDM effort will focus on cataloging these entities and assigning them universal keys that business applications can use to label these items so that they can interact with each other across the boundaries of systems and departments. Once these anchor points are established, the data models for specific types of products, organizations, and other core entities can be developed using an agile approach; creating new types of these objects or extending existing ones.

The key to getting the model right is to understand the characteristics of these universal elements and to build the initial model that can properly support those characteristics, which will vary little between companies or even industries. Take the "person" entity, for example; while there is little argument over what constitutes an individual person, many existing data models make the mistake of modeling "roles" (customer, employee, stock-holder, vendor contact, etc.) instead. A single person can be all of these, and more. Also problematic is the attempt to use external identifiers (e.g., Social Security number) as keys. Aside from the privacy issues, there is no universal identification system for living people, so you are best served with a surrogate key that carries no meaning and does not represent any sort of hierarchy. The person model also needs to handle synonyms, allowing you to note that two people one thought were separate are in fact one and the same.

Like the person entity, the "organization" entity must map to specific organizations, and not their roles. Unlike people, many organizations are collections of other organizations, and that is the way you want to manage hierarchies such as a department within a division. Most important, the model needs to account for multiple and overlapping hierarchies.

While it is tempting to view location as a set of GPS coordinates representing a particular point on our planet, this construct will not play well with the way location is defined in a business. The business definition of a location is almost always an area, not a point. Like the organization entity, there are locations that are collections of other locations, and these collections often overlap. Some locations are fairly abstract; for example, financial instruments are sold with settlement terms that name a particular exchange as the "location" even though the exchange is electronic and lacks any physical boundaries.

Of all of the core entities, the "product" entity will be the one most closely tied to the specific needs of the business. There are several common principles that apply in all cases: first, it's important to understand that the product domain should include the raw ingredients (flour, sugar, butter, water), the consumables (gas, electricity), the services (mixing, baking, cleaning) and the result (cookies). Like people, the scope of products is far broader than any specific product nomenclature, whether it is ISIN (for securities) or UPC (for consumer goods).

Also important is to carefully define what constitutes a distinct product. For that, I apply two rules: the first is whether one instance can be exchanged for another, without ANYONE getting upset; the second is whether the components of the product can be distributed without breaking anything. Using the second rule, a three-pack of printer cartridges would be a separate product from a single cartridge, since the blister pack would need to be broken.

Infrastructure can be understood as specific instances of products that are owned and used by organizations on an ongoing basis. Infrastructure elements typically have locations, and they are often assigned to people. Infrastructure also includes "intangible" assets such as patents, trademarks, and good will.

Because of their universality and their abstract nature, these core data models can be established quickly, without the need for lengthy review that normally accompanies an enterprise data model. Thereafter, the focus of the lean data management effort will be to grow the models and populate the repositories in support of specific business objectives, as well as to facilitate the use of these common keys by consumer applications. At that point, the ADM

approach can work quite well, with the consumer applications serving as stakeholders and perhaps the enterprise architecture office acting as the voice of the customer.

I welcome your comments on this issue of the Cutter IT E-Mail Advisor and encourage you to send your insights to [comments@cutter.com](mailto:comments@cutter.com).

- [Duff Bailey](#)

#### References

1. The benefits of MDM are well documented by Mark Fung-A-Fat (see "[MDM Success Through Better Customer Service](#)," 12 March 2008).
2. Agile MDM is described by Scott Ambler at [agiledata.org](http://agiledata.org).
3. Matt Ganis also provides some insights on how an EDM team can effectively interact with agile and non-agile teams (see "[Keeping the Peace: Mixing Agile and Waterfall Methods](#)," Cutter IT Journal, Vol. 20, No. 5).