

Master Data Management Meets SOA

A SYMBIOTIC RELATIONSHIP

WRITTEN BY JOHN KALOGIROU

➤ Master Data Management (MDM) is often defined as “management of master data (customer, product, supplier, etc.) that is shared across disparate IT systems and groups.” However, this simplistic description doesn’t do justice to the complexity of the MDM’s task and problem area. Master Data Management encompasses areas such as Customer Data Integration (CDI), Product Information Management (PIM), and Global Data Synchronization Network (GDSN); and partially overlaps the areas of Identity Management System (IdM), Business Intelligence systems, data quality, and data integration. This broad area of potential application causes multiple perspectives, diversity of stakeholders, and a fair amount of confusion across clients investigating an MDM solution.

The business need for MDM is made manifest both implicitly and explicitly. Its utility tends to be obvious in efforts around conformance and auditing, accurate reporting efforts, and a single view of the customer initiatives. However, MDM is often also a hidden requirement for successful consolidation projects after mergers and acquisitions. Its value in terms of return on investment, cost savings (reduced storage, reduced analysis, development, and maintenance, etc.), increased revenue (consistent view master data, reduced time to resolution, and effective decision making), and competitive advantage (operational efficiency, improved visibility to company performance, etc.) has been well documented by multiple reputable groups and authors (AMR Research, Forrester Research, Gartner, and the Yankee Group) so we won’t explore the existing benefits that the reader can easily reference. We will however discuss the benefits of MDM as they relate to SOA enablement.

MDM systems can be “federated,” “integrated,” or “hybrid” reflecting a combination of the first two fundamental architectures. These three types of system characteristics are as:

- **Federated MDM** – cross references key identifying information from participating systems to implement a registry-style solution. The main benefit of a federated solution is non-intrusiveness on participating systems that maintain their original context.
- **Integrated MDM** – stores all master data information from all

participating systems in a centralized MDM repository. This centralized repository houses the “gold copy” of all master data information. The main benefit of the integrated approach is that it provides the most complete, accurate, and consistent single view of master data.

- **Hybrid MDM** – stores common data elements from participating systems creating a “light gold copy” of the master data, while disparate elements are referenced from their original system of record. The benefit and drawback of the hybrid solution is the partial combination of the federated and integrated benefits.

Service Oriented Architecture (SOA)

From a systems design perspective, SOA is an architectural approach based on distributed computing principles. SOA has numerous other aspects in topics as diverse as business process design and IT governance. However, these aspects go beyond our scope here.

As an architectural paradigm, the participating components of a SOA system include: service providers, service consumers, intermediary services, and registries. A service provider publishes a service in the registry to be consumed by a service consumer who can identify the interface, purpose, and location of the service from the registry. Intermediary services intercept and handle operations that are common across services and can be leveraged instead of recreated every time. Typical intermediary services include: authentication, auditing, logging, monitoring, and message routing. All communications are done through commonly agreed on standards (UDDI, SOAP, WSDL, XML, HTTP/SSL). The design principles governing SOA are primarily object-oriented paradigms extended to address the service-oriented requirements. These service design principles include: loose coupling, service contract, abstraction, composability, autonomy, reusability, statelessness, and discoverability.

Services access information from a data services layer. A data services layer provides an abstraction layer between producers and consumers of data. The data services layer presents consumers with a virtual aggregated view of data from multiple data sources in a consistent and centralized fashion. The layer’s interface supports all consumers (human, application, external parties, or business services) while providing agility to data source providers.

A data service layer offers many benefits. Consumers are insulated from complexity, location, and changes in source data systems through abstraction. Providers have the flexibility to change underlying data schemas without impacting consumers through abstraction. Companies can centrally manage, monitor, measure, and report on the enterprise view of the data and metadata.

The three main categorizations of services in the data services layer are: Enterprise Data Services, Enterprise Metadata Services, and Enterprise Data Platform Services.

- The Enterprise Data Services area encompasses all the services around the data. For example, a request to be addressed by this area would be: Retrieve “gold copy” of customer “A” record.
- The Enterprise Metadata Services area includes all the services around the metadata. This area would address items such as: Retrieve master data schema of customer “A” record.
- The Enterprise Data Platform Services area supports all the services around the platform including management, monitoring, and reporting. An example of a request here would be: Retrieve MDM system, quality of service targets.

Services are defined in each area based on function (examples are shown in Figure 1). In each service and across all three areas, methods for search, access, create, update, delete, manage, monitor, and reporting functionality should be evaluated for applicability and realization.

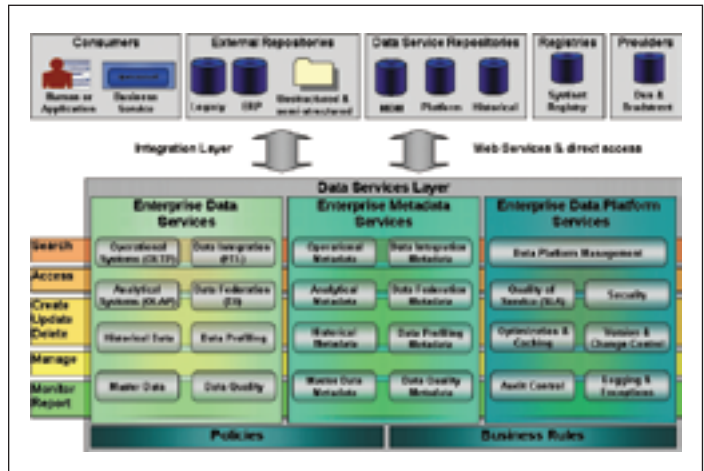


Figure 1: Data Services Layer

MDM Meets SOA

MDM and SOA evolved separately but share many design principles.

- **“Contract first”** applies to the interfaces in MDM and the service definition in SOA
- **“Reusability”** applies to data through conformance in MDM and services through SOA principles in SOA
- **“Discoverability”** applies to data through the master data repository in MDM and services through registry in SOA
- **“Abstraction”** applies to source system complexity and MDM and underlying service complexity under SOA.

MDM, however, typically doesn’t embrace SOA’s “loose coupling” principle. Extending MDM with loose coupling allows support for SOA’s semantic conformance needs.

As MDM practitioners contemplating supporting today’s SOA systems, we need to become familiar with SOA standards and strive for loose coupling with external systems. Eliminating point-to-point interfaces and replacing them with service-enabled integration minimizes the impact of changes from integration partners and consumers. Loose coupling should be applied internally as well to create an agile MDM system. An agile service-oriented MDM system provides its data quality, conformance, and other MDM functionality as business or data “services” available for net-enabled consumption by external parties. Finally, MDM systems should be able to handle the extensible data types (XML, HTML, PDF, and e-mail) common to net-centric application and be able to expose the master data model as part of the enterprise canonical data model (CDM) for service consumption.

Evaluate the maturity of your MDM system by answering the following questions:

1. Does my MDM support extensible data types such as XML?
2. Can internal, partner, or client services search, identify, and consume the CDM?
3. What is the effort for an application to participate in MDM as a consumer of the CDM?

4. What is the effort for an application to participate in MDM as a contributor to the CDM?
5. What is the effort to replace an external service provider?
6. Are data quality and conformance services exposed for use by applications or external parties?
7. What is the turnaround for changing the functionality of the current MDM system?
8. Can the MDM handle near real-time requests for conformance from participating systems?
9. Can the MDM exchange metadata with other MDM systems?
10. Can the MDM infer context and take action based on the semantics of the information being exchanged?

We've seen how SOA paradigms can contribute to the maturity of MDM. Now we'll focus on the other side of this marriage and see why SOA needs MDM. While SOA enables integration and data exchange through services, such integration is useless without a common vocabulary of data content and data structure. MDM defines how the enterprise establishes and maintains such vocabulary. To fully adopt enterprise SOA, an organization must first address MDM.

MDM is one of the most important components of the data services layer providing the required semantic integration of services for master data. Without an MDM system (or an ad hoc capability providing the MDM functionality), services don't know where to access the single version of the truth for "customer A" when there are multiple systems that capture "customer A" information. Moreover, this 'customer A' information has to be the same in terms of structure, as well as content, when it's consumed by services.

The technical intersection of MDM and SOA occurs at the data services layer. For the data services layer to provide consistent information to consumers across the multiple data providers, data and metadata inconsistencies, discrepancies, omissions, and overlaps have to be addressed. This means that MDM functionality must be present. MDM crosses and has elements in all three areas of the data services layer mentioned above.

- Service enablement of traditional MDM functionality such as data quality and data harmonization (synchronization of data across participant systems and MDM) is exposed under the Enterprise Data Services area.
- The SOA designers and developers who are creating business services, as well as others consuming services, have to reference the organization's master data schemas. These master schemas are exposed in the Enterprise Metadata Services area, allowing consumers to draw inferences directly from their semantics.
- Finally, in the Enterprise Data Platform we find services around the management, measuring, monitoring, and reporting of the MDM system.

When the topic of semantically conformed data is raised, many firms point to their data warehousing initiatives. Consider the following typical question in that regard:

"I don't have MDM but I do have an existing Enterprise Data Warehouse (EDW) that covers the integration and data quality that seem to overlap what MDM is supposed to do. Can't I service-wrap my existing system to achieve SOA MDM benefits?"

Service-enabling the EDW is indeed a worthwhile and beneficial initiative but doesn't offer MDM capabilities. The reason is that most prior EDW efforts have concentrated on the integration and "business view" of data from disparate sources rather than the harmonization of the data at the source systems – meaning that when the source systems are service-enabled, they'll be in semantic conflict with the service-enabled EDW. In addition, EDW typically doesn't concentrate on master data, master metadata, and related data governance issues. Therefore, most EDW systems don't have the faculties for managing master data (such as an interface with workflow for business users to manage master data through its lifecycle). Finally, to successfully proceed towards an MDM solution, the initiative must be executive sponsored and business-owned versus an IT service enablement of an existing application project. To be clear, an existing EDW should, can, and will be leveraged in the new data services layer but doesn't replace the need for an MDM system.

Evaluate the maturity of your data services layer by answering the following questions:

1. Are my services impacted by changes to the repositories or databases being accessed?
2. Do my services have to call two or more repositories to read or update information?
3. Do two or more of the repositories being accessed contain overlapping information?
4. Do my analysts and developers need to understand the system internals and entity models for each interfacing system?
5. Do we have duplicate, incomplete, missing, or conflicting data across systems?
6. Can the data service layer provide a "single version of the truth" for the master data (customer, product, vendor, employee, and assets)?
7. Are my services semantically integrated to define the customer, product, or vendor?
8. Would data quality and conformance intermediary services be helpful?
9. Does my service taxonomy include services to access data and metadata across providers?
10. Is my enterprise data model exposed for consumption by the services?

The quest for the modern MDM system, and the single (data) version of the truth for SOA enablement, entails many challenges and risks. The highest business risks across both types of initiatives include: lack of executive support, insufficient business (versus IT) data ownership, and inappropriate skills or expertise. From a technical perspective, the highest risks include: inadequate performance, incorrect security of the exposed enterprise model, and ill-advised vendor selection.

Marrying MDM and SOA makes sense and offers vital benefits from both the MDM and the SOA perspective. Expect to see more such paired initiatives and vendors repositioning their products to cover the joint space better. ■

About the Author

John Kalogirou is MomentumSI's information management director. John has 15 years experience in managerial and technical roles guiding SMBs and Fortune 500 companies to implement information, integration, and intelligence solutions that improve business effectiveness and profitability.