Editor:
Office of the CTO
Architecture Excellence

Author:
Wolfram Kleis

SAP Architecture Bluebook

Master Data Management

Version 1.1

December 2007

Internal

History

Date	Version	Name	Change/Enhancement
2007/07/30	0.9	Wolfram Kleis	Released internally for SAP
2007/08/20	0.91	Wolfram Kleis	Corrected spelling errors
2007/09/06	1.0	Wolfram Kleis	Corrected spelling errors and wording in chapter 5
2007/12/18	1.1	Wolfram Kleis	Corrected errors in Figure 8 and section 5.2.2

Acknowledgement

The following colleagues contributed in various ways to make this or previous editions of the document possible: Christian Behre, Rüdiger Buck-Emden, Eugene Cherny, Nicolai Jordt, Aaron Mahimainathan, Holger Meinert, Gordon Mühl, Frank Samuel, Rainer Schäfer, Michael Theis, Thomas Vogt and Paul Weinberg.

Table of Contents

Table of Figures

1 Abstract

Master data management (MDM) is defined as activities, rules and mechanisms that ensure the availability and quality of core enterprise data which rarely changes and is used by many business processes. Typical master data quality issues are incorrect data as well as unmanaged redundant data with unclear identities and inconsistent attributes. These problems can be addressed by data cleansing (removing errors and deduplication) and by defining responsibilities and processes for maintaining master data.

In distributed application landscapes master data management helps enterprises to improve integrated processes across application systems, and to improve enterprise wide reporting and analytics. Master data management can be done in a non-central mode between single systems or in a centralized way, typically using an architecture with a central MDM hub. There are different ways to do centralized master data management, which differ with respect to the questions where master data is maintained and how consolidated master data is used and distributed.

The SAP NetWeaver MDM product comprises the MDM technology and MDM specific business content. The SAP NetWeaver MDM technology supports centralized master data management in different styles from one-way consolidation for read-only use to central maintenance of master data.

The central component of SAP NetWeaver MDM is the MDM server which provides access to the central MDM repositories. The MDM server is a standalone server with optional integration with the SAP NetWeaver Application Server, which uses its own in-memory database architecture. Application systems typically exchange master data with the MDM server in a loosely coupled style using the SAP NetWeaver Exchange Infrastructure (XI) and file based communication.

The user interface for administrators and MDM experts is provided by a set of client tools. Business users can use the MDM Portal UI to search and maintain master data from within the SAP NetWeaver Portal. Most of the functionality of the MDM server is exposed via APIs (ABAP, Java and COM). Basic interfaces for searching and maintaining master data are exposed as Web services. The functionality of the MDM tools comprises system specific inbound and outbound mapping of data structures and values, master data cleansing, mass data maintenance, and data validation based on business rules.

SAP NetWeaver MDM comes with its own workflow component, which allows for defining master data management processes and for integrating external data enrichment and validation services. SAP NetWeaver MDM also provides tools and connectivity for creating and publishing catalog content.

2 Introduction

2.1 Goal and Scope

The goal of this document is to give an overview of master data management architectures in general and of the SAP NetWeaver MDM product in particular. The document describes the status quo of May 2007 based on SAP NetWeaver MDM 5.5 SP5. Target group are readers inside SAP who are interested in a technical introduction to master data management and the SAP NetWeaver MDM product. Chapter 3 of the document gives an introduction to MDM concepts and architectures in general. Chapter 4 lists general requirements for MDM solutions. Finally chapter 5 gives an overview of the architecture and the functionality of the SAP NetWeaver MDM product (based on SAP NetWeaver MDM 5.5. SP5).

2.2 Motivation

Master data is the core data of an enterprise that exists independent of specific business transactions and is referenced in business transactions. Typical examples are customer data, vendor data, employee data and product data. Master data is a critical asset of an enterprise and the quality of master data is important for optimized business processes and accurate reporting. Master data management processes that ensure master data quality and availability are required even in a single application system. Additional problems arise due to the fact that SAP customers usually have distributed landscapes with multiple systems that create and store master data locally. The reasons for distribution are for example the use of dedicated systems for different business functions (multiple ERP systems, CRM, SRM, SCM), the use of separate systems for different geographical locations and also the fact that customers often run systems from different vendors. With data spread across systems additional problems such as global consistency, physical replication and semantic and technical mapping have to be solved. Master data management has the goal to improve the quality of master data by installing processes for master data governance and by providing tools for data cleansing. In addition it enables the sharing of master data across applications by providing the required mappings and data exchange mechanisms.

Master data management solutions are advertised as the foundation for improving integrated processes and as a prerequisite for accurate company wide reporting. Master data management is also often named as one of the foundations of an enterprise service architecture. Services from different applications that can be integrated into one business process require a consistent common view of underlying data. For SAP applications master data management is important for example for the integration of the components of the SAP Business Suite or for realizing flexible deployment options in AP.

3 MDM Concepts and Architectures

3.1 Definition of Master Data and MDM

3.1.1 What is Master Data?

Before we discuss master data management we should first try to define what master data is. However, the reader should be aware that there is no formal definition of master data and the border between master data and other data is fuzzy.
Data is typically categorized as transactional data describing a single business event or as reference data which is referenced in many business events. Master data is reference data that has some special characteristics.

- In [1] master data is defined as reference data that is never or infrequently changed.
- The definition in [2] puts more focus on processes and policies: Master data is defined as agreed upon central resources that can be shared across systems and for which special policies and procedures for creation and maintenance are defined.
- In [3] Gartner defines master data as *"the consistent and uniform set of identifiers and extended attributes that describe the core entities of the enterprise and are used across multiple business processes"*.
- The SAP terminology database SAPTERM defines master data as:
  "The fundamental data your company requires about an individual, organization, or product. Master data is used regularly by your company, is presented in a consistent manner, and rarely needs to be changed.[1]"

There seems to be a common understanding that master data is the core data of an enterprise, with high data quality requirements, often shared across applications and infrequently changed.

Some examples for typical master data objects are data for employees, customers, vendors, products, accounts or locations. The differentiation between master data and non master data is sometimes defined not on object level but on attribute level. The same object (or database table) may contain attributes that are master data and attributes that are not categorized as master data.

For SAP applications the term "master data" covers much more than the few object types listed above. For AP/A1S more than 130 business objects were classified as master data. To give the reader an impression what master data means in SAP applications, a list of proposed AP/A1S master data objects is included in the appendix (see 9.1). Note also that depending on the SAP product some master data objects are technically treated as customizing data (for example currencies, countries, bank master records).

3.1.2 What is Master Data Management

Gartner defines master data management as a *"process in which business units and IT departments collaborate, cleanse, publish and protect common information assets that must be shared across the enterprise. MDM ensures the consistency, accuracy, stewardship and accountability for the core information of the enterprise"* [3].

[1],see https://portal.wdf.sap.corp/go/sapterm results for "master data" (SRD-MD)

This definition makes clear that master data management is not just a technology. Master data management is an enterprise wide task that requires the cooperation of business units and which includes policies and responsibilities for managing master data.

We can derive an alternative definition of master management by stating its goals: The main goal of master data management is to *ensure the availability of master data in good quality*. Master data management is the sum of all activities, processes and technical mechanisms that are required to achieve this goal.

But what are the risks that are threatening the quality of master data? Master data quality problems are incorrect data (such as spelling errors, non existing addresses) and multiple but unrelated records that represent the same real world object. If the common identity of these records is not discovered, this will often lead to wrong results, for example in reporting or when negotiating conditions with vendors or customers. When multiple representations of the same real life objects are discovered, they must be managed, for example by avoiding semantically different attributes. The term “master data cleansing” is sometimes used to describe only the correction of errors. In this document we use it in a broader sense that subsumes the correction of errors, the detection of multiple instances of the same real world object, and removing the inconsistencies between their attributes.

Multiple instances of the same real world entity are called duplicates or identicals depending on whether they exist in the same system or in different systems:

- Duplicates are multiple redundant representations of the same real life object that exist unintended in the same system. The goal of de-duplication is to detect duplicates and manage them. Eliminating the duplicates physically is often not an option because the duplicate instances are already used in transactions and referred to in business documents. Managing duplicates means to keep track of the sets of objects that have a common identity and to ensure that the common attributes and relationships have consistent values in all instances.
- Identicals are representations of the same real life object that exist in different systems. Identicals are often required. When information about the same real world object is needed by processes in different systems, it has to represented in each of them. Therefore identicals are not a problem as such. The goal of master data management is to manage the identicals. This includes knowing which objects correspond to each other in the different systems and ensuring that their attributes are consistent across systems. Unmanaged identicals may lead to duplicates if the data is replicated between systems without the required identity mapping.

Master data management is required, no matter which IT landscape is used. It is required even in a single system. Duplicate master data records, for example, may be created in the same system, simply because an employee creates a new record for an already existing customer. And even with a single system it is required to define responsibilities and processes for maintaining master data.

With distributed system landscapes, master data management becomes more difficult and additional technical problems arise. In distributed landscapes, master data management is required to create a single and accurate enterprise-wide view on relevant master data. This is for example needed for company wide business analytics. In addition, cross-system business processes must be enabled by providing a common and consistent interpretation of the referenced master data. This includes an agreement on common data semantics, the mapping of identifiers and the mapping and transformation of attributes.

3.2 Distributed MDM Architectures

3.2.1 Master Data Ownership and Routing

Master data management in a distributed landscape includes the definition of rules that specify

- Data ownership: In which systems should a particular category of master data be created and maintained?
- Data routing: To which systems should a particular category of master data be distributed?

Master data management solutions may differ with respect to the degree of system support for defining those rules. If there is no system support, the rules must be managed manually and compliance must be enforced by conventions and organizational measures.

The rules for master data ownership and routing can be defined on the level of business object types, on the level of data attributes, or on instance level (based on values). In the latter case, the rules depend on actual values of attributes. With value-based routing it is for example possible to distribute master data of American employees to one system and those of European employees to some other system. Subscription-based distribution mechanisms can be used that allow subscribing systems to master data that matches a set of criteria based on type and content.

3.2.2 Degree Of Coupling

An important property of a distributed application landscape is the degree of coupling between the application systems. The required degree of coupling depends on the scenario and is an important input for selecting the right master data management architecture.

3.2.2.1 Tightly Coupled Systems

With tightly coupled systems, master data is validated and exchanged immediately using synchronous communication. When new master data objects are created as a part of a business process, the data is immediately validated, enhanced and distributed. Tight coupling enables the immediate prevention of inconsistent and undetected identicals by executing synchronous checks for existing objects before new master data objects are created. The problem with tight coupling is that each system depends on the availability of the other systems. In a tightly coupled federation of systems, all participating systems have to be permanently available.

Tight coupling often also means that the participating systems communicate more frequently and exchange a higher amount of data compared to the loosely coupled alternative. This can lead to scalability problems if a bigger number of systems has to be connected in a tightly coupled way.

The participating systems must be designed in a way that enables the tightly coupled cooperation (by providing service interfaces, interception points etc.). Tight coupling is normally not an option for connecting legacy systems or third party systems.

3.2.2.2 Loosely Coupled Systems

Loosely coupled systems communicate asynchronously. They often – but not necessarily - exchange master data less frequently and synchronize only parts of the master data. Loosely coupled systems work in an autonomous way and are able to operate even if the other systems are temporarily not available. Because of the problems enterprises faced with tightly coupled systems (availability, scalability) in the past, today loosely coupled systems are state-of-the-art in enterprise information technology. Enterprise service architectures are networks of loosely coupled systems that communicate via asynchronous messages. Therefore a master data management solution for enterprise SOA scenarios must support loose coupling.

In the context of MDM, loose coupling means that master data is exchanged asynchronously. Unless central master data maintenance is used (see 3.2.7.4), the global validation and cleansing of master data is also done asynchronously. In that case data quality problems (such as incorrect data or inconsistent duplicates and identicals) are initially tolerated and removed or compensated later.

Loosely coupled master data management architectures are typically chosen to connect systems that belong to different organizations (or sub organizations), or to connect legacy systems or third party systems. For master data management tasks such as data cleansing, the asynchronous communication style is often the best choice, because human interaction is required to resolve data conflicts or merge duplicates.

Tightly coupled systems can use a loosely coupled mode as a fallback solution. In a tightly coupled MDM scenario, a system could by default call a service synchronously for creating new master data objects in a remote location. If the remote service is not available, the system could switch to the loosely coupled mode and create the data locally. This way the business process would be able to continue without blocking, but with the risk that global inconsistencies might be introduced.

3.2.3 Process Steps For Exchanging Master Data Between Application Systems

The typical process for exchanging master data between a source system and a target system is as follows: (“source” and “target” are roles that are defined for one specific exchange of data.)

a) Collect and extract relevant master data in the source system

b) Perform structural mapping

c) Perform value mapping including mapping of keys (identifiers)

d) Physically transport the data

e) Validate inbound data based on rules valid for the target system

f) Complete the inbound data with local attributes and relationships that are mandatory in the target system. This is required in case only a part of the local master data is shared between systems. Incoming master data records are typically incomplete in the context of the target system and must be augmented locally to match with the local constraints. (Note that this step is optional. In a replication scenario it is not required because in that case the import of data in the receiving system is completely automated and the sender has to provide all the necessary data).

g) Optionally perform manual approval steps based on the policies of the target system

h) Persist the data to local storages (tables) in the target system

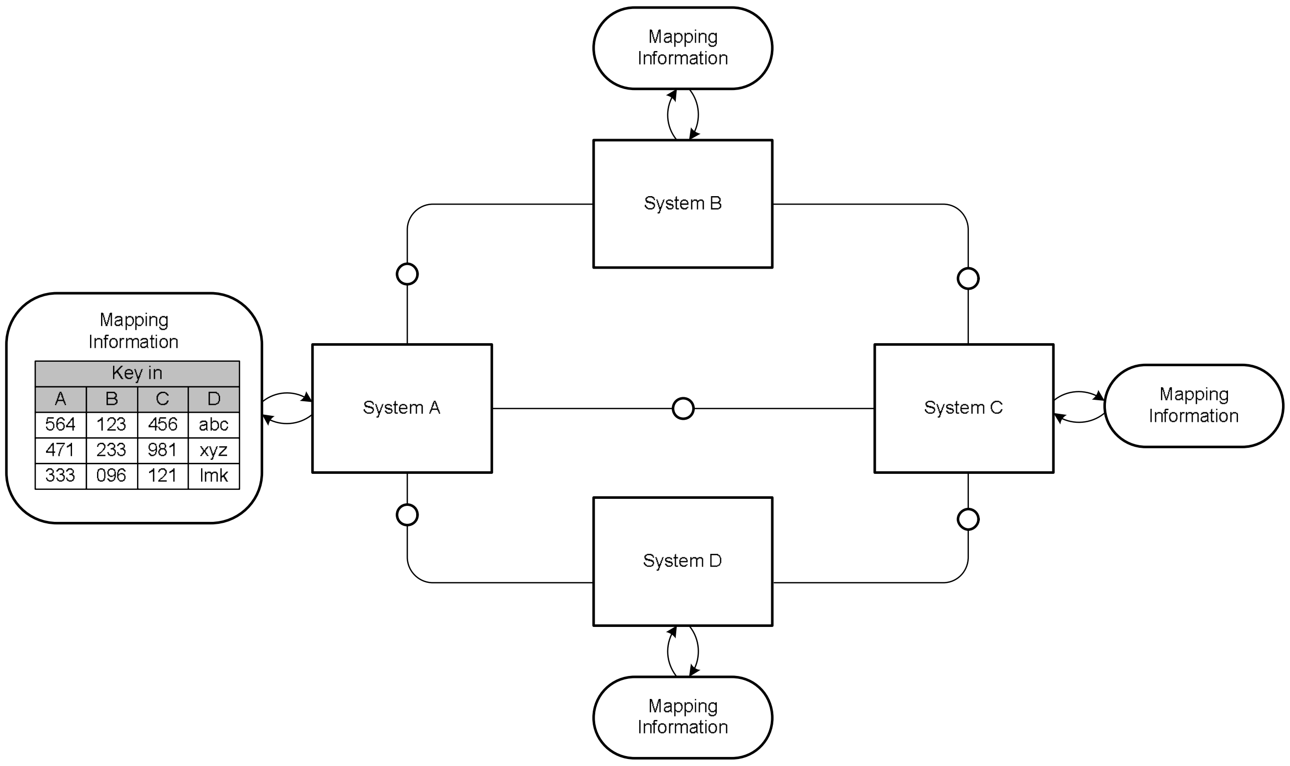
Note that the above list does not imply a temporal ordering. It is also not defined where the different steps are executed. The mappings, for example, can be done in the source system (before the physical transmission), in the target system (after the physical transmission) or by some component in between. It is also possible that the mapping is divided into steps that are executed at different locations.

Based on the alternatives for deploying the described steps, two architectural approaches for master data management can be distinguished: The peer-to-peer approach and the centralized approach. These are discussed in the following sections.

3.2.4 Peer-To-Peer MDM

With the peer-to-peer approach two business application systems exchange master data directly without a central MDM infrastructure (Figure 1). This requires a kind of “bilateral MDM” between the two communicating systems: The required mappings and transformations are done by the communicating systems or by communication middleware that is used to establish the channel between them. In addition, local key mapping solutions can be used. This is the traditional approach in which systems talk to each other using remote APIs, messages or services. Even though there is no central MDM infrastructure in use, this is still an example for doing MDM: Master data is mapped, transformed and exchanged. Having an agreement about which system is responsible for creating a specific type of master data, there is not even the danger of creating inconsistent or undetected identicals (master/slave scenario).

The peer-to-peer approach is one option for sharing master data in order to enable cross-system business processes. However, this non-central approach is not sufficient to cover all master data management requirements. The goal to provide a single enterprise wide view of master data by definition requires some kind of centralized structure. Either a specialized central master data management component is introduced or one of the existing systems has to play that role. Detecting and removing global inconsistencies in a distributed and heterogeneous landscape is also much easier when there is a central MDM component.



Key in			
A	B	C	D
564	123	456	abc
471	233	981	xyz
333	096	121	lmk

Figure 1 Example of a Peer-to-Peer MDM Architecture

3.2.5 Centralized MDM Architecture

A centralized MDM architecture typically uses a central MDM hub with a central MDM repository. The MDM repository stores at least information about master data, such as key mappings transformation rules and data quality rules. It often also physically contains the shared master data. Alternatively the data can be stored in a federated way with the central repository storing only meta data (see 3.2.7.2). Connected to the MDM hub are two types of systems:

- the operational systems where master data is used for business transactions
- data warehouse and analytical systems where master data is used for analytics and reporting.

This setup is shown in Figure 2.

It is important to point out that the diagram in Figure 2 shows only the communication paths for the *management* of master data. The centralized architecture with the MDM hub is used for master data management related communication and for providing central MDM services that can be consumed by the application systems. The MDM hub should not be mistaken as the central communication hub. Having a central MDM architecture does not mean that all master data must be exchanged via the central MDM hub. Usually additional direct channels exist between the systems. Those channels are used for direct communication while executing a business transaction. Via these channels, both master data and transactional data can be exchanged. This setup is shown in Figure 3.

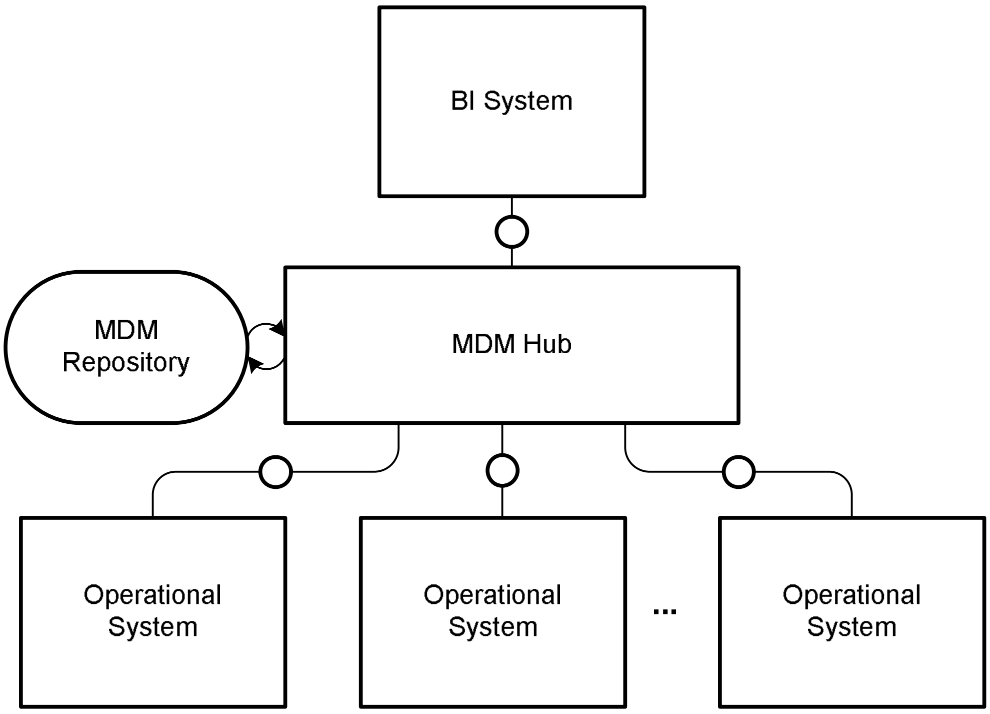


Figure 2 Centralized MDM Architecture

The topology in Figure 3 can be classified as a hybrid solution that mixes the peer-to-peer and the centralized architectures. When a business process is executed and direct communication with some other system is required, this is done using a direct channel, which requires some local mappings and transformations of master data. The central MDM infrastructure can be used for an asynchronous cleansing and synchronization of master data at a later point in time. The central MDM hub can also provide validation services that ensure the global quality of master data. Those validation services can be called by the application systems before initiating a business transaction.

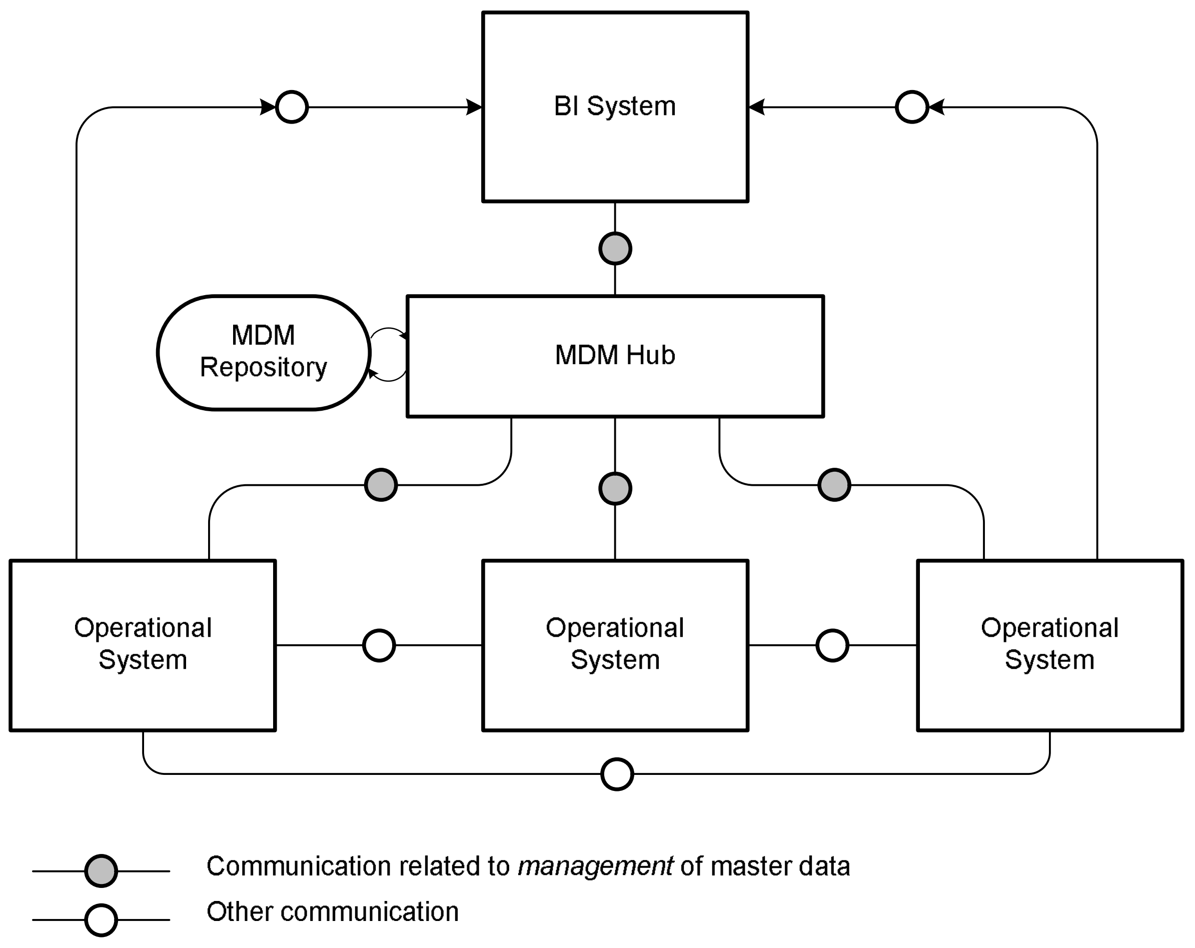


Figure 3 Channels for MDM and Channels for other Communication

3.2.6 Delta Mechanisms For Distributing Master Data

For every kind of data distribution task it makes sense to optimize the communication by transmitting only information that is not yet existing in the target location. This can be achieved by using a delta mechanism, which means that only the changes are sent that were made since the last transmission. Delta communication is not a topic that is specific for master data management. It is a general optimization pattern that must be considered for MDM as well.

For implementing a delta mechanism, some knowledge about the state of the recipient is required (or at least some valid assumption). With only one sender and a reliable channel the delta communication can be implemented quite easily: If a sender can assume that all relevant information was distributed to the recipients at some point in time, it is sufficient to send from then on only new changes, creations and deletions. If the sender can assume that the updates are always received successfully, the sender only needs some basic change tracking and the information when the last updates were sent to the different recipients.

With multiple senders, the delta management becomes more difficult. Even if sender A never sent a piece of information to recipient B, the information may be already there via some other sender. A strictly centralized architecture with a central hub makes it easier to implement delta processing. The systems that send data only have to keep track what was already sent to the hub. For distributing the data the central hub is the only sender, which makes it easier to determine the deltas that have to be sent to the different systems.

Delta implementations also become difficult if the channels are not reliable or if the communication partners can “forget” information. To tackle these problems, a more sophisticated protocol for delta communication may be required: sender and receiver first have to exchange meta data to clarify which data are really missing in the target system. In addition, delta management requires conflict resolution capabilities at the receiving side (for example the possibility to request resending of a record). There is a trade-off between the costs for delta management one side and the savings that can be achieved on the other side.

An in-depth discussion of delta processing is beyond the scope of this document. For the purpose of this document it is sufficient to have some basic awareness of the problem and to understand that delta management is especially challenging with non central architectures.

3.2.7 Options For Centralized MDM Architectures

3.2.7.1 SAP's Classification of MDM Architectures

MDM Architecture	System of Origin	Use of Central Master Data	SAP IT Scenario
One-way consolidation	Operational systems	Read only use of consolidated master data (analytics, catalogs, ID mapping provisioning for peer-to-peer communication) No distribution back to operational systems	Master Data Consolidation
Two-way synchronization		Distribution to all attached systems (local completion in operational systems)	Master Data Harmonization
Single system of origin	Central MDM Repository or dedicated operational system	Distribution to all attached systems (local completion in operational systems)	Central Master Data Management

Table 1 SAP's Classification of Centralized MDM Architectures

SAP classifies MDM architectures with respect to the questions "Where are the master data created/updated? and "How are the consolidated master data used and distributed"? The answers to these questions lead to three MDM architectures which SAP describes as MDM IT scenarios (Table 1).

3.2.7.2 One-Way Consolidation

The first architectural option is shown in Figure 4. Master data is created and maintained in the operational systems. From there it is exported using extractors. The extractors comprise two things: A mechanism to define which objects and attributes are relevant for master data management, and the functionality to collect and export the data.

The extracted master data are transferred to the MDM hub where they are consolidated into the central MDM repository. The consolidation process includes the detection and resolution of conflicts. Conflicts are for example unmanaged duplicates or violations of global business rules that define allowed attributes and relationships. The rules and mappings for consolidation and global validation are also stored in the central MDM repository. In this architecture, the MDM hub is used as the system of record for all globally relevant master data.

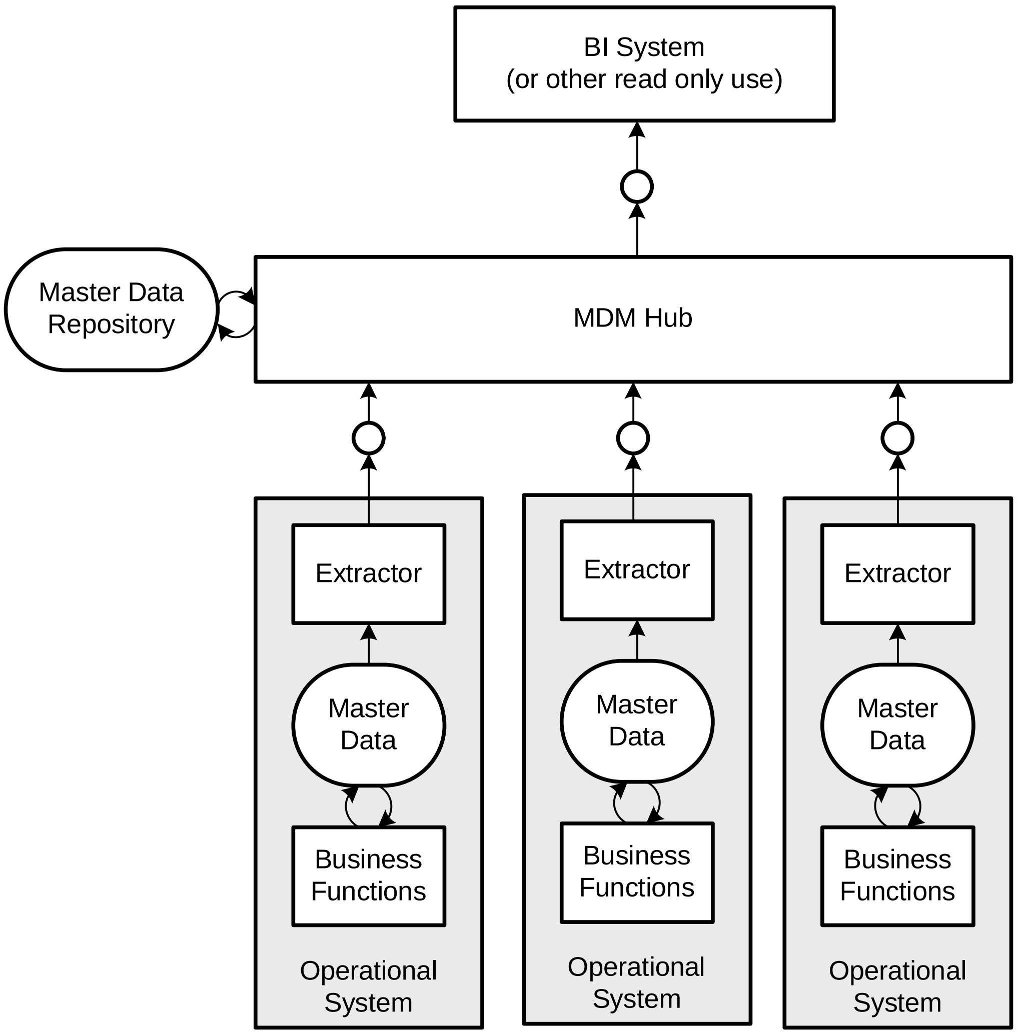


Figure 4 One-way Consolidation

Storing the global parts of master data of an enterprise in a central repository requires a common data model that must be agreed upon by the different parties involved. This data model often contains a common core that is relevant for all applications and additional parts that are only relevant for selected parties. The definition of the central data model is a non-trivial task and requires cooperation of technology experts and business experts of different areas.

With the "one-way consolidation" architecture, consolidated master data is not distributed back to the application systems. The central view is used for read-only purposes, for example by feeding the data into a BI system. Global spent analysis is a typical example that requires consolidated master data. The publication of a catalog based on consolidated product data is another example.

When the central MDM architecture is set up, it is required to initially gather the data from the operational systems and load them into the master data repository. From then on, only new and changed records are extracted and consolidated (delta mechanism).

The "one-way consolidation" architecture supports consistent company wide reporting, analytics and publications. As no master data are fed back into the operational systems, this architecture does not help to improve the quality of the data used in the business transactions. This architecture is sufficient if only the consolidated view is required and if it is not necessary or not feasible to affect the application systems.

A variation of this setup is a federated architecture: The master data is not physically copied into a central repository. Instead the master data hub keeps only a master data registry that stores the global identifiers of the master data and the mapping to locations in the application systems that store the actual data. The difference between a master data repository and a master data registry is discussed for example in [4].

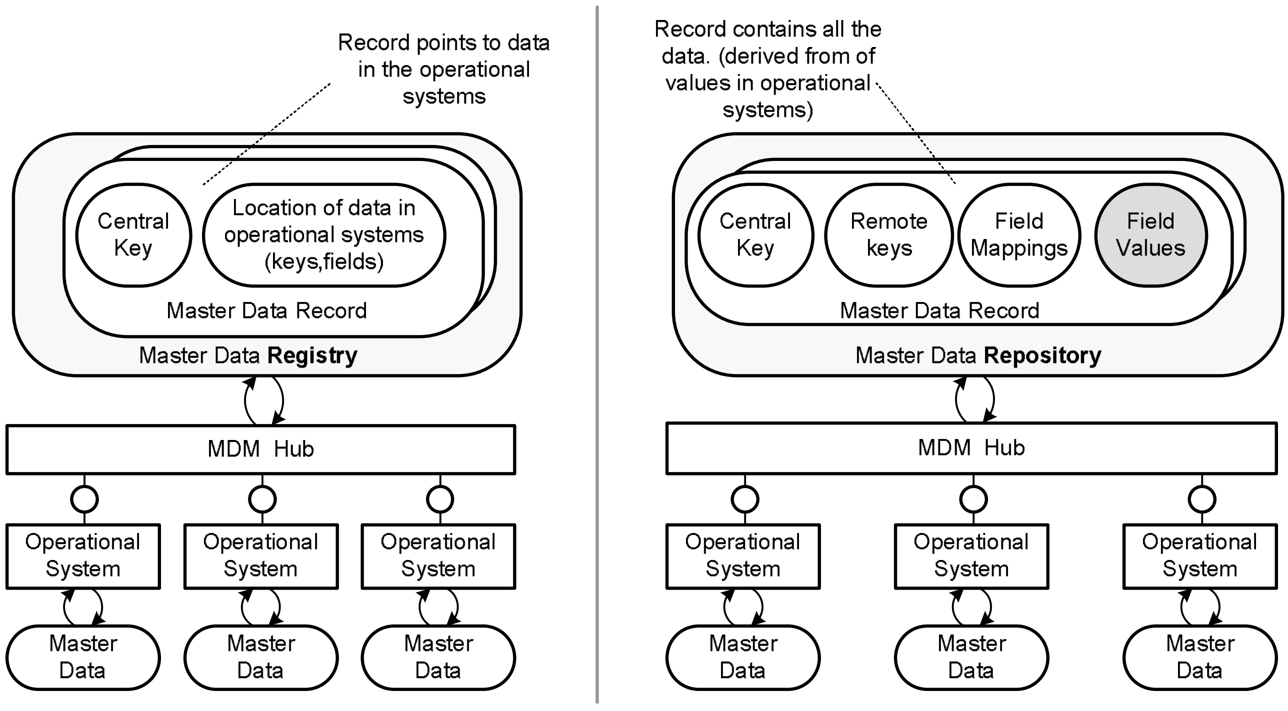


Figure 5 Master Data Registry versus Master Data Repository

On the left hand side of Figure 5 the registry variant is shown. The MDM hub stores only central keys and the information which attributes of which objects in the remote systems belong to a master data object. The single view of the master data is put together dynamically on demand from the different sources. The registry variant is sometimes also called “virtual system of record”. It has the advantages that the data need not be imported physically into the central hub and that there are no centrally stored redundant data to be kept in sync. This alternative is not offered by SAP. Instead SAP offers the repository variant which can be seen on the right hand side of Figure 5. The repository variant has an MDM hub that stores the complete master data including the field values. The repository style has the advantage that the central copy of the data can be used to cleanse master data without affecting the operational systems. Another advantage is that the central view is available physically. A real time enabled federated query technology is not required and the availability of central master data is not depending on the availability of the operational systems.

3.2.7.3 Two-way synchronization

Figure 6 shows the second architectural style: The operational systems are the systems of origin where master data is created and maintained. It is typical for this scenario that master data of the same type may be created and maintained in multiple systems. Master data is collected from different sources into a central repository where data cleansing is applied. The consolidated state is then distributed back into the operational systems. This scenario requires a distribution mechanism for master data and components in the operational systems that are able to accept the incoming consolidated master data. In Figure 6 the operational systems contain an agent called “integrator” which represents the functionality needed for integrating inbound data. In this context integration refers to the inbound validation, local completion of data, local approval, and distribution of data to local storages. The operational systems also contain master data of two categories: Shared master data which is synchronized with other systems and master data objects and attributes that are relevant only locally. The integrator agent has modifying access to both categories of master data: The shared master data is written when distributed data records are imported. The only locally relevant data is modified when imported master data is completed with locally required attributes.

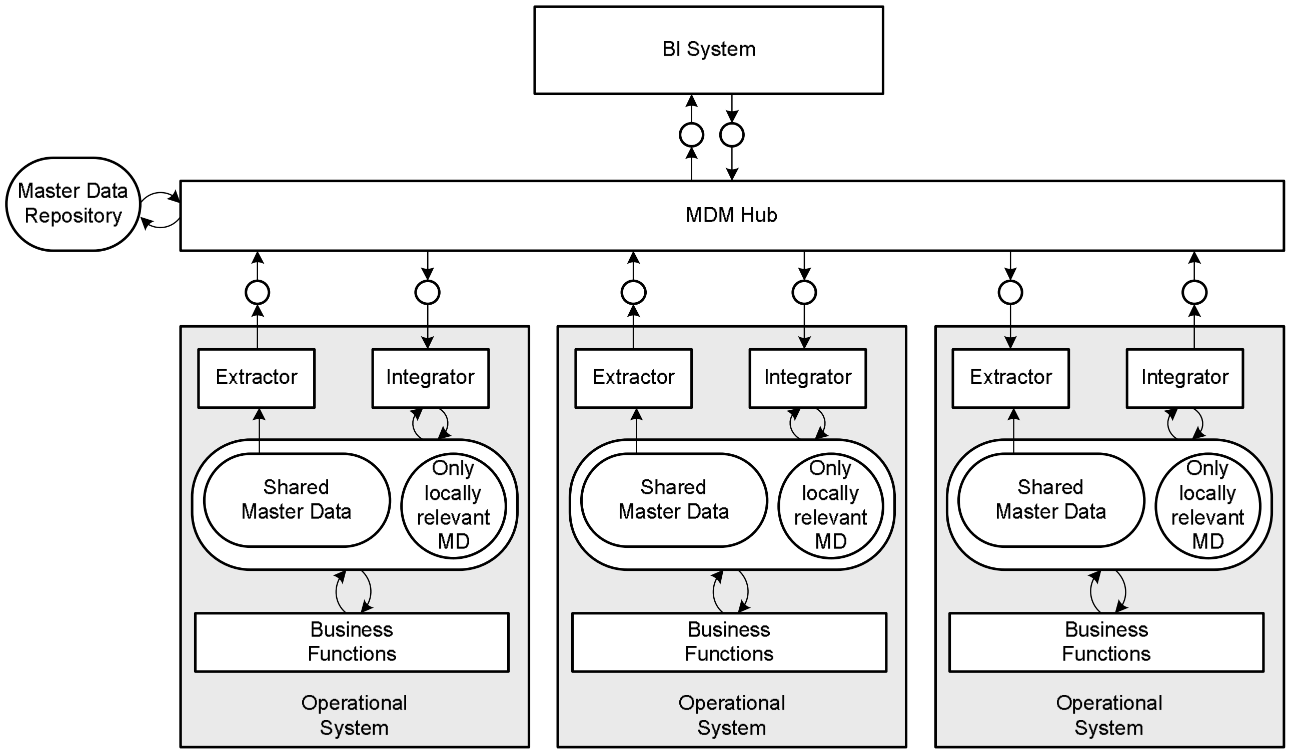


Figure 6 Two-way Synchronization

When duplicates or identicals are detected during data consolidation, one has to either choose one of the instances as the "survivor" or a new record has to be created in the master data repository. The repository must keep the mapping information that describes how the new record (or the selected survivor) maps to the records in the application systems (key mapping). As explained above duplicates are typically not deleted but managed. The goal is to ensure that the globally relevant attributes of the duplicates have consistent values in all systems.

Like in the "one-way" scenario we can distinguish two phases

- Initial consolidation and distribution
  Relevant master data are extracted from the source systems, loaded into the master data repository, cleansed and distributed to all target systems.
- Ongoing consolidation and distribution of deltas (new, changed or deleted data)

The export/consolidate/import functionality available in this architecture can be used for a special use case: The migration of master data. This is necessary when an old system is replaced by a new one and there is no data import/migration functionality provided by the vendor. The migration requires an initial consolidation of master data from the old system and then an initial distribution into the new system. For master data synchronization it is often sufficient to exchange only the global part of the master data. Note that this is different for migrations. The migration scenario requires that all data is transferred to the hub.

3.2.7.4 Single System of Origin

The third architectural variant is shown in Figure 7. With this architecture, a single system of origin is defined for all types of relevant master data. This single system of origin is typically the central MDM repository. In that case the entire lifecycle of a master data record is under central control. Rules and policies can be defined and enforced centrally. Duplicates and inconsistent or incomplete data can be detected and corrected at the time of creation.

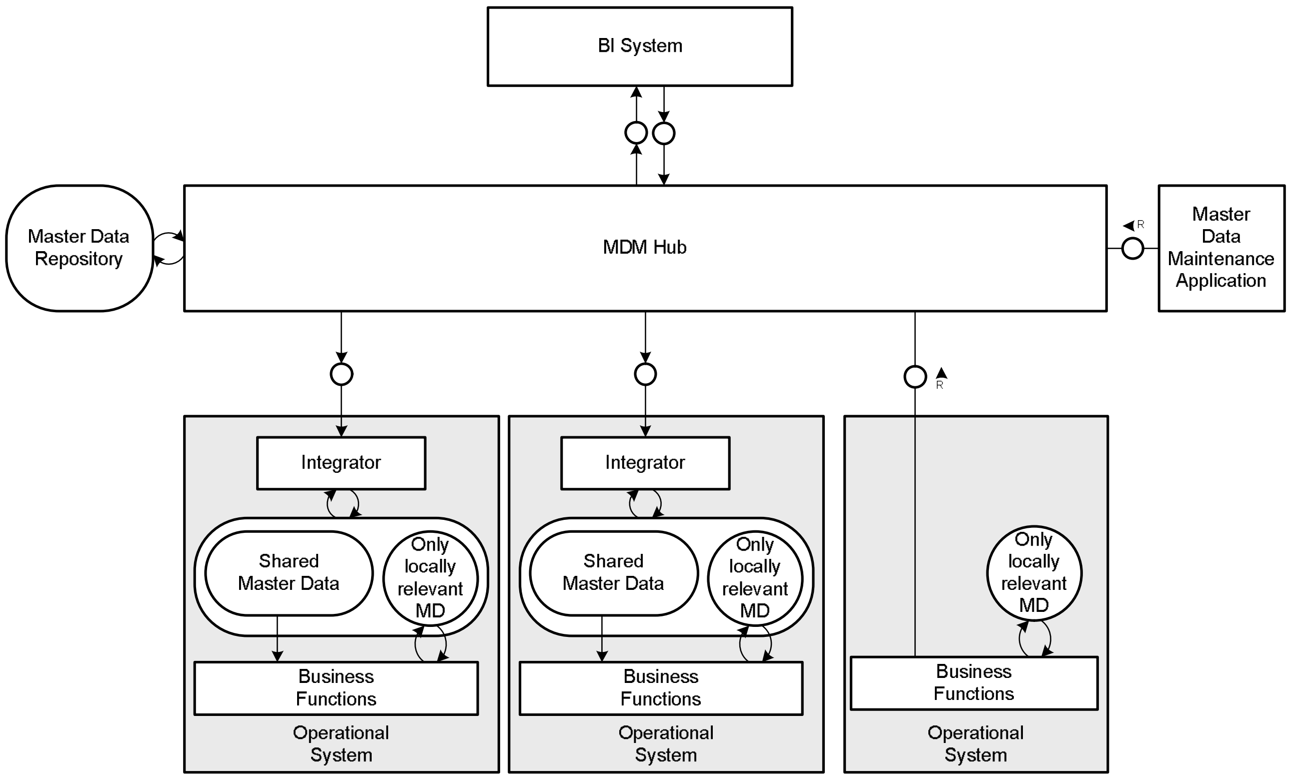


Figure 7 Single System of Origin

Figure 7 contains an agent named “Master Data Maintenance Application” which represents the tools that allow users to maintain master data in the central repository. This agent sends requests to the MDM hub which then updates the central repository. The MDM hub has access to the rules stored in the repository and can therefore enforce the global quality of master data. It is not relevant in which system the “Master Data Maintenance Application” itself is executed. This is basically a user interface and it could be located on the central MDM hub, in a separate system and in the operational systems.

Figure 7 also shows two alternatives for accessing shared master data in the operational systems: The systems on the left hand side and in the middle keep read only copies of the centrally managed shared master data. Maintaining centrally managed master data is disabled in that system. The operational system on the right hand side works differently. That system was configured to read and write master data always by calling services of the MDM hub (assuming that such a configuration option exists). No copies of centrally managed master data are stored in that system. The advantage of this alternative is that there is no redundancy and therefore no synchronization effort. The disadvantage is the high load that is placed on the central system and the complete dependency on its availability. This alternative is listed here only for completeness. It is not relevant for real life scenarios with SAP systems: SAP application systems always store the master data locally and the same decision was made for the AP application platform.

Central master data management with a single system of origin is the most challenging alternative:

- When a piece of master data is created centrally, it must be validated based on the business rules of the application systems for which these data are relevant. This means that the validation rules and the data context required to evaluate these rules must be available for the central master data maintenance process. There are two solutions for this problem:

  - One solution is to replicate the entire data context into the central repository and to re-implement all relevant business rules on the central MDM hub. In many cases this may not be a viable solution because of the high amount of data and the additional effort to duplicate business rules and keep them up to date. It would also require that the technology used for describing central validations must be powerful enough to express all relevant rules.
  - An alternative solution is a distributed validation architecture that allows the MDM hub to call the business application systems as external rule valuators.

- Upon import into the operational systems, the centrally created master data often must be enriched or completed with additional attributes and relationships before they can be used. This additional process step must be implemented in or on top of the operational systems. (This also applies to the two-way synchronization scenario in 3.2.7.3).
- The single system of origin must have the ownership for master data. Existing functionality in business systems that would allow local modification of centrally managed master data must be routed to the single system of origin or switched off. Ideally this should be technically configurable in the involved systems. Otherwise it must be ensured by organizational rules.
- The single system of origin is required for all processes that modify or create master data. Therefore a high level of system availability is required.

It is not required that the single system of origin is always the central hub. As a variant of the pattern it is also possible to use a particular application system as the system of origin for a specific type of master data. For example an ERP system could be defined as the single system of origin for financial master data. In that case maintaining financial master data would not be allowed in all other systems (including the MDM hub). To guarantee the quality of global pieces of master data, the ERP system must either perform all required checks or its must be enabled to call validation services of the central MDM hub before it accepts new or changed master data. Such a setup would still be of type “single system of origin” but with different systems for different categories of master data.

Note that the different styles of central MDM architectures described in this section can be combined: Different operational systems can be integrated in different ways and different approaches could be used for different types of master data.

3.2.7.5 Gartner’s Classification of centralized MDM Architectures

In addition to the classification presented above, other classifications for MDM architectures are possible. Gartner, for example, classifies MDM architectures into three architectural styles [3]. This classification is based on the questions how master data are physically stored and whether a central hub is used in transactions or not. Gartner defines 3 architectural styles for centralized MDM which are listed below and are mapped to the styles we have defined above.

- **Registry style**
  This is the ”virtual system of record” described in 3.2.7.2. The MDM hub does not store the actual master data but only a registry of global IDs and meta data.
- **Coexistence style**
  The consolidated view of shared master data is stored in the central MDM hub and distributed to the other systems. This way master data is harmonized across the system landscape. The MDM hub is used for collecting, reconciling and distributing. The harmonization of master data happens asynchronously in batch mode “after the event”. Because of this latency, the centrally stored master data are not used in real time transactions. Compared to the classification in 3.2.7.1 this style maps best to the “two-way-synchronization” style.
- **Transaction Hub Style**
  The consolidated master data is stored in the central MDM hub. The MDM hub offers SOA services that are directly used by business transactions in the operational systems. Compared to the classification in 3.2.7.1 this style is a variant of the “single system of origin” that provides services for accessing consolidated and up-to-date master data in real time. The transactional hub style can be

applied when developing new applications. It does not fit for old releases or systems from vendors that have no integration with the services of the MDM hub.

3.2.8 Comparison Of The MDM Approaches

	"Peer-to-Peer" MDM	Central MDM
Advantages	Master data are mapped and transformed with full knowledge of the concrete business context. The data and the mappings are interpreted from the view of the specific business application. It is not required to change the architecture of the business applications, for example for calling validations on a central hub.	Provides single system of record (physically or virtually) for consolidated master data Provides single system of origin (optional) Enables central control, policies and governance With the MDM repository as the common reference model only N mappings must be defined to connect N application systems. Enables central tracking and auditing of MDM activities Easier to implement delta management
Disadvantages	The mappings and rules for handling master data are distributed across the systems and often not separated from the logic of the business process There is no way to centrally define rules for data quality and MDM processes In the general case, the non-central architecture is not able to produce a single global view of master data Difficult to implement delta processing because of multiple senders Without a common exchange format, N(N-1)/2 mappings must be defined to connect N systems.	Central data model for master data must be agreed upon and developed. Central checking of rules often requires the duplication of business logic and raises the question of missing data context on the MDM hub. Mapping master data for exchange between systems may require application context which may not be available on a central hub. Application systems must be enabled to participate in central MDM. High availability and performance required for central hub (bottleneck, single point of failure) if master data maintenance processes are business critical. User interfaces for maintaining master data may have to be re-implemented

Table 2 Peer-to-Peer versus Central MDM Architectures

4 Requirements for MDM Products

In this chapter we list the most important requirements for MDM products[2]. Note that not all these requirements need to be fulfilled in every case. Based on the customer scenario a subset may be sufficient.

An MDM product consist of two parts: The first part is a software system or infrastructure that supports MDM functions and processes. The second part are the scenarios, business content and processes as well as the integration with business applications that are MDM enabled.

4.1 Common Requirements

These are the requirements that should be fulfilled by any MDM solution, no matter whether it is a central or a non-central approach.

1. Mechanisms for creating the enterprise wide consistent view on master data (”single version of the truth”) must be provided.
2. Mechanisms are required for defining and enforcing responsibilities and policies for master data management.
    - Which is the process for creating or changing data of a specific type?
    - Who is allowed to view, create, update or delete which type of master data?
    - In which location (system) are different categories of master data created and changed (data ownership)?
    - It should be possible to define a role based MDM workflow that allows to define review and approval steps.
3. It must be possible to exchange master data between systems. This includes mechanisms for export, import and mapping.
4. It must be possible to define and enforce rules for data quality. This includes detecting and managing duplicates and identicals, ensuring completeness of data, and the validation of master data related business rules.

4.2 Requirements For Central MDM Hubs and Repositories

If the decision was made to use a centralized architecture, the following additional requirements must be considered:

1. Data modeling capabilities are required that allow to define the central master data model of the enterprise.
    - The central repository must be able to describe different types of master data needed in the enterprise.
    - It should be possible to describe relationships between master data, including hierarchies.
    - Especially it must be possible to model all pieces of master data that are relevant for SAP business applications.
    - Customers must be able to extend data models for example by adding new attributes.

[2] See [1], section 2.5 and [3], pg. 6

2. Efficient mechanisms are required for collecting and distributing consolidated master data
   - Guaranteed delivery of consolidated master data between systems or between systems and MDM hub
   - After an initial distribution, only deltas should be transferred.
   - It must be possible to distribute creations, updates and deletions of master data records.
   - It must be possible to define - on attribute level - which parts of the central master data are distributed to which target system (master data routing).
3. Mechanisms are required for distributing MDM meta data (schemas, rules, mappings) especially among different MDM hubs (test landscape and productive landscape).
4. The solution must be application independent. The master data management solution should be applicable to all kind of master data independent from the specific application or domain.
5. Change tracking and versioning: It should be possible to track which changes were made by whom in which system. All activities such as creations, updates, imports, distributions, merging of records should be recorded and traceable. This is an important requirement to ensure auditability. In addition, versioning enables the system to provide historic views of the data. In the one-way consolidation and the two-way-synchronization scenarios (no central master data maintenance) identicals are merged into one master record but the original records may still be changed afterwards. In that case versioning and tracking of merge relationships is needed to re-evaluate the merged record.
6. The MDM hub should be enabled for participating in an enterprise service architecture. Master data and the provided MDM functionality should be accessible via enterprise services. Process integration and automation of MDM functionality should be enabled by corresponding interfaces.
7. Ability to process mass data within a reasonable time: Depending on the size of the enterprise and the area of business, the system must be able to deal with millions of master data records. In particular the system must be able to process the initial filling with data from connected systems in reasonable time.
8. The product should support different architectural styles (see 3.2.7)

Requirements that are covered by SAP product standards (security, performance, availability, etc) must be fulfilled by every SAP product and are not listed here explicitly.

5 The SAP NetWeaver MDM Product

This chapter gives an overview of the architecture, technology and features of SAP NetWeaver MDM (based on the SAP NetWeaver MDM 5.5 SP5 release).

As stated in chapter 4 an MDM product consists of MDM software on one hand and scenarios and content on the other hand. Accordingly the SAP NetWeaver MDM product consists of two parts, the MDM technology (MDM repository, MDM server, client applications, APIs) on the one hand and business scenarios and content on the other hand.

5.1 MDM Content and Business Scenarios Shipped By SAP

5.1.1 MDM Business Content

MDM business content includes MDM repository structures, validation rules, predefined inbound and outbound mappings, XML schemas , SAP NetWeaver XI content, extractors for different SAP systems and portal content. For extractors SAP NetWeaver MDM only defines the extraction framework. The actual extractors are provided by the applications.

Currently SAP ships content for the following types of master data and SAP application systems:

Master Data Type	**SAP Application**
Material Master	SAP ERP, SAP R/3
Customer Master	SAP ERP, SAP R/3
Vendor Master	SAP ERP, SAP R/3
Employee	SAP ERP, SAP R/3
Business Partner	SAP SRM, SAP CRM, SAP R/3
Product	SAP SRM, SAP CRM, SAP R/3
Article	SAP Retail

Table 3 Supported SAP Application Systems and Content

5.1.2 MDM Business Scenarios

In addition to the object types listed above, SAP NetWeaver ships and documents three MDM business scenarios.

The *Rich Product-Content Management* scenario is about centralized management of product catalog content and multi-channel publishing.

Global Data Synchronization (GDS) is a special scenario for exchange of trade-item information between manufacturers of consumer packaged goods and retailers using a certified data pool and a global registry. This requires to map SAP ERP material data to trade items understood by the data pool infrastructure, maintaining information about trade partners and exchanging messages with trade partners. To support the GDS scenario the product includes GDS business content (MDM repository, mappings), adapters for the communication with the data pool and a GDS client application. GDS is a special scenario because it deals with external models and infrastructure for data exchange between organizations. A description of the GDS solution is beyond the scope of this document. This is the reason why components such as the GDS client and the connection to data pools do not appear in the architecture models presented in section 5.2.

The *Customer Data Integration* scenario allows to collect, cleanse and re-distribute customer data from and to various systems (SAP ERP, SAP CRM and non SAP).

In addition to these three scenarios there is the *SRM-MDM catalog* scenario which is a part of SAP SRM. This scenario is used to populate the product catalog in SAP SRM with content from an MDM repository.

5.1.3 MDM IT Scenarios

The SAP NetWeaver MDM platform allows to implement a centralized MDM solution with an MDM server and MDM repository as the central hub. It supports the three architectural styles presented in 3.2.7 in the non-federated variant that physically stores consolidated master data in a central MDM repository. Most of the SAP NetWeaver MDM customers start with the *master data consolidation* IT scenario (one-way consolidation) and then move on to the *master data harmonization* IT scenario (two-way synchronization).

The *central master data management* IT scenario (single system of origin) is currently not implemented by most customers, although many customers see this scenario as a goal for the future.

Out of the box, the SAP NetWeaver MDM product supports loosely coupled integration based on asynchronous file-based exchange of master data. This basically matches with Gartner's MDM "coexistence" style MDM architecture (3.2.7.5). A more tightly coupled integration can be achieved as a custom-made solution using the programming interfaces offered by SAP NetWeaver MDM.

5.1.4 SAP NetWeaver MDM as a Generic Technology

SAP NetWeaver MDM was designed as a generic technology that can be used by the customers to implement their own MDM solution. This solution depends on the actual needs, processes and system landscape of the customer. A turn-key MDM application would very likely not fit with the customer requirements. The generic approach promises the required flexibility: The MDM platform allows to implement different architectural styles and it also provides APIs for implementing the customer specific parts. For the most common cases the scenarios are predefined and shipped by SAP with the corresponding integration and content.

5.2 SAP NetWeaver MDM Architecture Overview

The block diagram in Figure 8 shows the major components of the SAP NetWeaver MDM technology and how they are connected[3]. SAP NetWeaver MDM is based on a client-server architecture with MDM repositories, an MDM server and a variety of client components. User interfaces are provided for administrative task, repository design, and for working with master data. The user interfaces are offered either by standalone client applications or inside the SAP NetWeaver Portal (see 5.2.2).In addition there is an export/import architecture for importing and distributing master data, an MDM workflow component and a set of APIs and services. All these components are covered in subsequent chapters. As shown in Figure 8 there is also an optional integration with the CCMS (Computing Center Management System) for real time technical monitoring and with the System Landscape Directory (SLD) to add MDM to the inventory of components that are installed in the customer landscape.

[3] For complete list of installable components see the software component matrix in the SAP NetWeaver MDM Master Guide (see [6]).

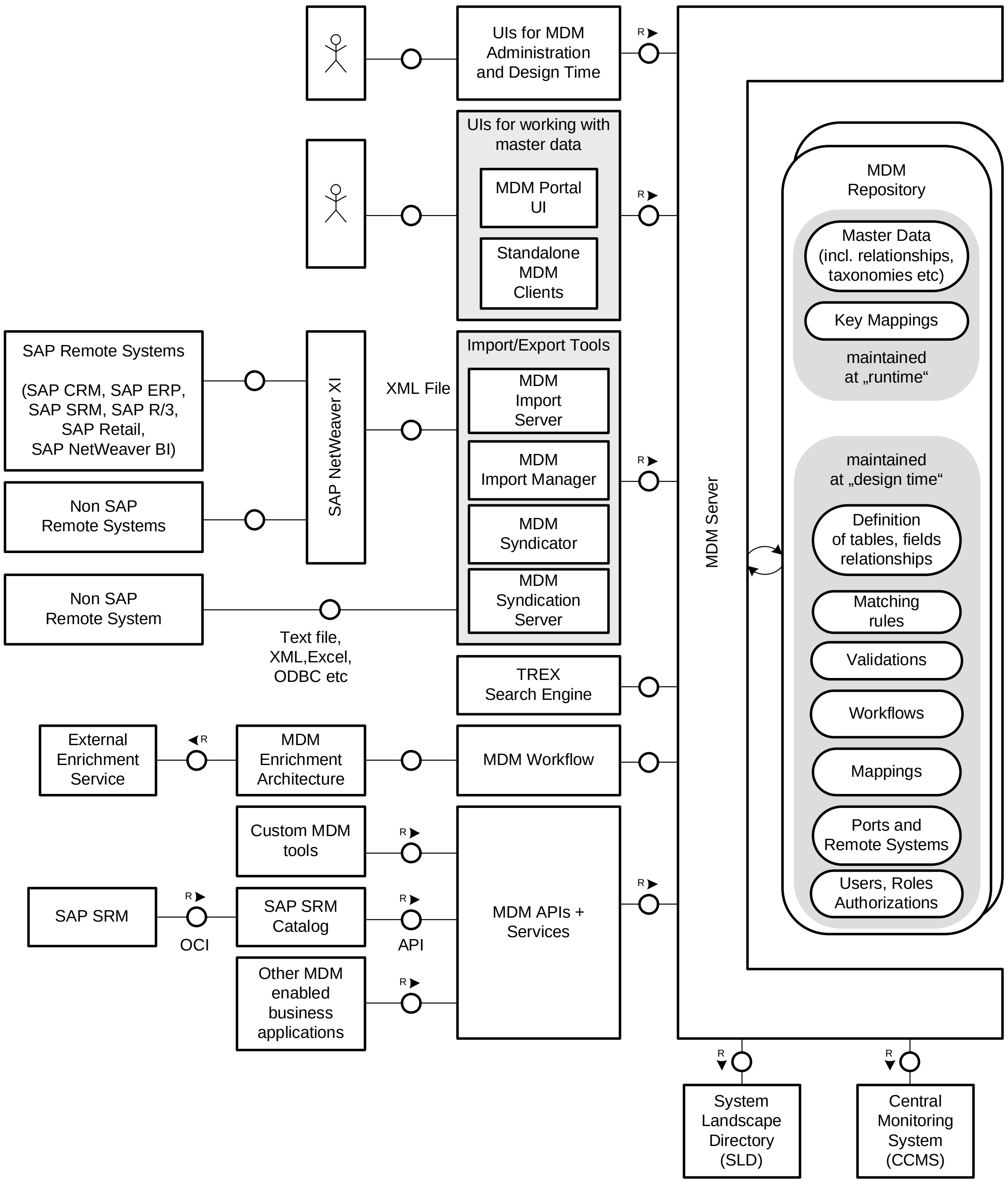

UIs for MDM Administration and Design Time
UIs for working with master data
MDM Portal UI
Standalone MDM Clients
SAP Remote Systems (SAP CRM, SAP ERP, SAP SRM, SAP R/3, SAP Retail, SAP NetWeaver BI)
Non SAP Remote Systems
Non SAP Remote System
SAP NetWeaver XI
XML File
Import/Export Tools
MDM Import Server
MDM Import Manager
MDM Syndicator
MDM Syndication Server
Text file, XML,Excel, ODBC etc
TREX Search Engine
External Enrichment Service
MDM Enrichment Architecture
MDM Workflow
Custom MDM tools
SAP SRM
OCI
SAP SRM Catalog
API
Other MDM enabled business applications
MDM APIs + Services
MDM Server
MDM Repository
Master Data (incl. relationships, taxonomies etc)
Key Mappings
maintained at „runtime“
maintained at „design time“
Definition of tables, fields relationships
Matching rules
Validations
Workflows
Mappings
Ports and Remote Systems
Users, Roles Authorizations
System Landscape Directory (SLD)
Central Monitoring System (CCMS)
R

Figure 8 SAP NetWeaver MDM Architecture

5.2.1 MDM Server And Repositories

5.2.1.1 MDM Repository

On the right hand side of Figure 8 the MDM Server is shown which provides access to the MDM repositories. Each MDM server can access one or more MDM repositories. An MDM repository contains two categories of data: data that must be defined “at design time” before the productive use can start and data that is created “at runtime” by the MDM users during normal operation. The data that is created during normal operation comprises the actual master data (including records in master table and lookup tables, hierarchies, taxonomies and relationships as explained in 5.3) and the key mappings that are required to manage the relationship between duplicates and identicals.

The following data that must be created at design time, before productive use can start

- the repository schema including the definition of tables, fields and relationships
- transformations, rules and strategies for matching records (finding duplicates and identicals)
- data validation rules
- workflow definitions
- mappings for import and export
- definitions of ports and remote systems (see 5.4.1)
- users, roles and authorizations

Separate MDM repositories are required for different types of business objects. There is for example one repository for customers and a separate one for products. More details about the data model of the MDM repository can be found in 5.3.

5.2.1.2 MDM Server Architecture

The MDM server contains core MDM functionality such as the engines for searching, matching and validating. The MDM server is a standalone application implemented in C++ which is not running inside the SAP NetWeaver Application Server.

Figure 9 shows a refined view of the MDM server and the MDM repositories which is closer to the implementation. The MDM repositories are stored in relational databases. One MDM server can access multiple MDM repositories that may be persisted on one or on multiple database servers. It is also possible to store a single MDM repository using multiple (predefined) physical partitions that are stored in separate databases. This feature can be used, for example, to store rarely changed product images in a separate database with different database backup schedules.

Inside the MDM server there are the core MDM services that provide the functionality and a database abstraction layer that encapsulates the details of different relational database management systems.

To achieve high performance the MDM server uses an in-memory cache into which complete MDM repositories are loaded. Before a repository can be accessed by clients, it must first be loaded into the cache of the MDM server by an administrator. The in-memory architecture is required to enable the fast searching and matching provided by the MDM server. But it also has an impact on hardware requirements and server operation: The server hardware needs enough main memory to completely load all repositories assigned to that server. Loading of very big repositories takes considerable time which must be added to the startup time after a server restart or repository maintenance.

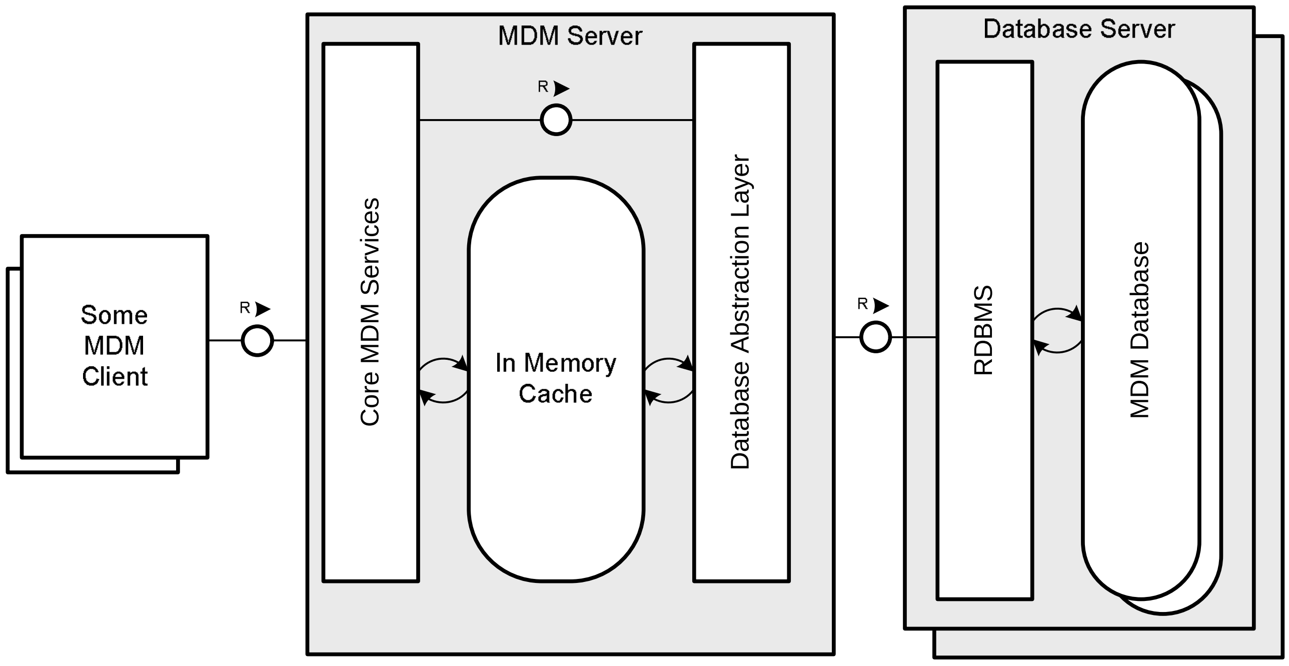


Figure 9 MDM Server and MDM Databases

For maintaining repository metadata (such as data model, remote systems or ports), the repository must be unloaded from the memory cache. This means that the repository is not available for users, which may become a problem if a schema maintenance takes a long time. The solution for this problem is to use a separate repository for developing schemas, mappings etc. When the development is finished and tested, the definitions are exported from the development repository and then are imported into the production repository. The production repository would be unavailable only for the short amount of time needed for the import.

5.2.1.3 Slave Repositories

For improving read performance and as a failover strategy there is the concept of slave repositories. A slave repository is created as a read-only copy of an MDM master repository. Users can access the slave repository only for reading. The slave repository can be updated from the master repository either manually in the MDM Console application (see 5.2.2.1) or by scheduling a script that calls the MDM command line client. Slave repositories can also be used to improve read performance by assigning different groups of users to different slave repositories that are synchronized with the same master repository. In addition a slave repository can be used as a fallback system by defining it as the new master repository in case of emergency.

5.2.2 MDM Clients and MDM Portal UI

Users can access the MDM functions using the MDM portal UI or a variety of client applications for the different MDM activities (see Figure 10). The MDM Import Manager and the MDM Syndicator are client applications that are used for importing and exporting master data and for defining the required mappings. They are covered in detail in 5.4. Some clients that are specific for the publishing scenario (such as the MDM Indexer or MDM Image Manager) or for the GDS scenario (see 5.1.2) are not included in Figure 10. The same is true for utilities such as the MDM language selector. Refer to the SAP NetWeaver MDM Master Guide [6] for a complete list of MDM clients.

5.2.2.1 MDM Console

The MDM Console is a Windows client application which is used by two types of users.

- Administrators use the MDM Console to start and stop MDM servers, to load and unload repositories into and from the server's memory cache and for user management.
- Data Model developers use the MDM console to design the structure of a repository. This activity comprises the definition of the data models, roles and authorizations and the definition of import and export connections to the different remote systems.

The same functionality is also offered by the MDM CLIX command line client. This can be used for scripting automated solutions and as an interactive client on non-Windows platforms.

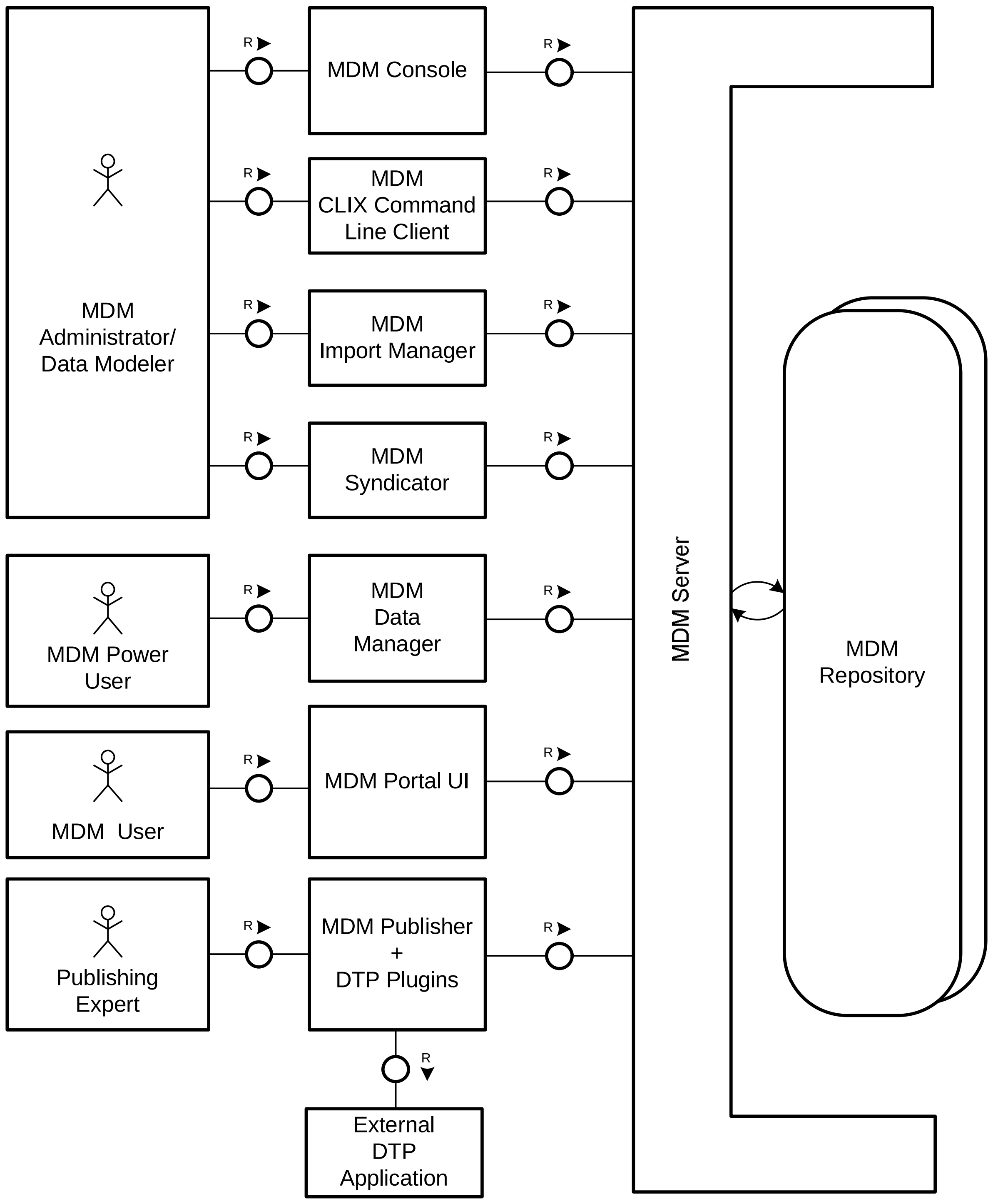


Figure 10 MDM Clients and UIs

5.2.2.2 MDM Data Manager

The MDM Data Manager is the Windows GUI client for working with the master data stored in an MDM repository. It is used for:

- maintaining master data in the repository
- browsing and searching
- defining validation rules and assignments (see 5.6.2)
- matching master data records, defining matching transformations, rules and strategies, finding and managing duplicates in the repository.
- Defining and executing the MDM workflow (workflow details are defined using Microsoft Visio)

The MDM Data Manager with its capabilities for defining rules and matching strategies is a tool for power users.

5.2.2.3 MDM Portal UI

Master data search and maintenance functionality for the standard user are in addition offered in the MDM Portal UI. With the MDM product a set of portal iViews is shipped that implement this functionality. To use this UI an SAP NetWeaver Application Server with SAP NetWeaver Portal is required in addition to the MDM server.

5.2.2.4 MDM Publisher

For the Rich Product-Content Management scenario there is the MDM Publisher client which is used to create catalog layouts that are stored in the repository and can be further processed in desktop publishing applications.

5.2.3 Data Import and Distribution

In the center of Figure 8, the architecture for importing and distributing master data from and to remote systems was shown. This part is refined in Figure 11.

The MDM Import Manager and the MDM Syndicator are interactive applications that are used to define the parameters for import and export such as mappings, transformations and filtering (see 5.4). They can also be used to interactively import and distribute data. The MDM Import Server and the MDM Syndication server are used for automated import and distribution.

The usual way for connecting remote systems to the MDM server is via SAP NetWeaver XI. Master data is extracted from the remote systems and then sent via SAP NetWeaver XI. If required, SAP NetWeaver XI transforms the messages into XML format and places them in defined location in the file system (using shared network folders or FTP). From there the files are imported into the MDM repository (see 5.4.2 for details).

The outbound direction works in a similar way: The master data records that are to be distributed are transformed to the target structure and stored in an XML file in a well defined location. The XML files are picked up by SAP NetWeaver XI and delivered to the target system. In addition to the import and export via SAP NetWeaver XI there is also the option to directly connect to a database system via ODBC or to directly import/export plain text files or Microsoft Excel documents from/to the file system. Details about the export architecture are described in 5.4.3.

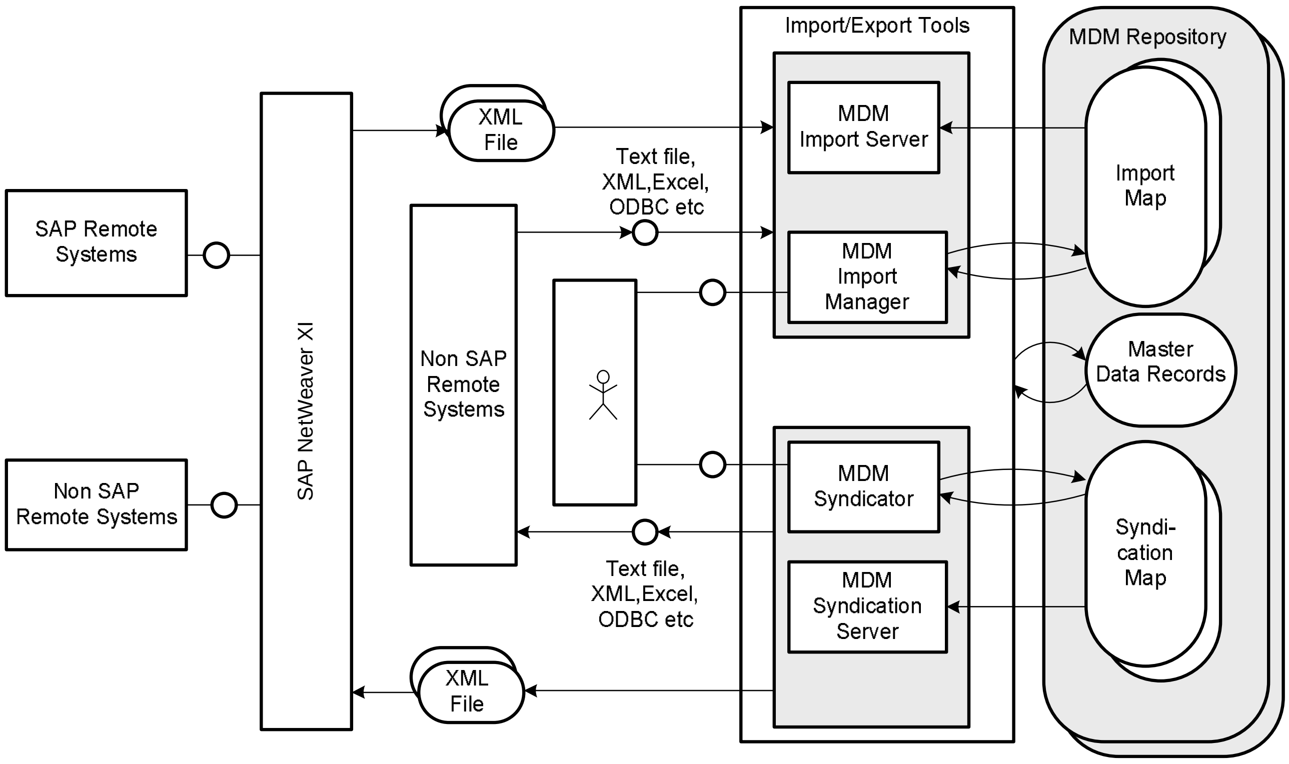


Figure 11 SAP NetWeaver MDM Architecture for Import, Export and Distribution

5.2.4 TREX Integration

Figure 8 also shows that there is an integration of the MDM server and the TREX search engine. After installing an MDM plug-in on the TREX server, TREX can be used to search PDF documents that are stored in the MDM repository.

5.2.5 MDM Workflow

The MDM Workflow component in Figure 8 is a workflow engine for defining master data management related workflows. The workflow items appear either in the MDM Data Manager or – via portal integration – in the universal work list in the enterprise portal. Workflow definitions are initially created in the MDM Data Manager, but this means just to define the workflow name and some basic parameters. For defining the workflow steps Microsoft Visio is used with an SAP NetWeaver MDM specific extension. As shown in Figure 8 external data enrichment services can be contained as steps in an MDM workflow using the enrichment architecture. See section 5.5 for details.

5.2.1 MDM APIs

The overview diagram in Figure 8 contains a block named “MDM APIs and Services”. The inner structure of this block is refined in Figure 12.

The refined diagram shows the various interfaces that can be used to access MDM functionality from applications:

- The MDM COM API is used for developing Windows applications.
- The Java API can be used to develop standalone Java applications or Java server applications.

- The MDM ABAP API can be used to access the MDM server from ABAP applications. For better integration and performance, the MDM ABAP API is implemented directly in the kernel of the SAP NetWeaver Application Server.
- The APIs differ with respect to the supported operations. The most complete API is the Java API which gives access to the full set of functionality of the MDM server. The set of tasks for which interfaces are exposed includes searching and updating master data, accessing key mappings, server and repository management, triggering syndications, matching duplicates, schema definition and creating and running validations.
- There is also a set of MDM Web services available. There are generic Web services for managing key mappings and for searching, creating and updating master data records. Generic services are required because new business object types can be invented by SAP and by customers at any time. In addition to these generic services there are specific services for searching, creating and updating objects of a specific type (business partner).

The APIs can be used to integrate application systems with MDM or to develop custom MDM tools that operate on the MDM repository. All API libraries access the MDM server using a MDM specific network protocol based on TCP/IP.

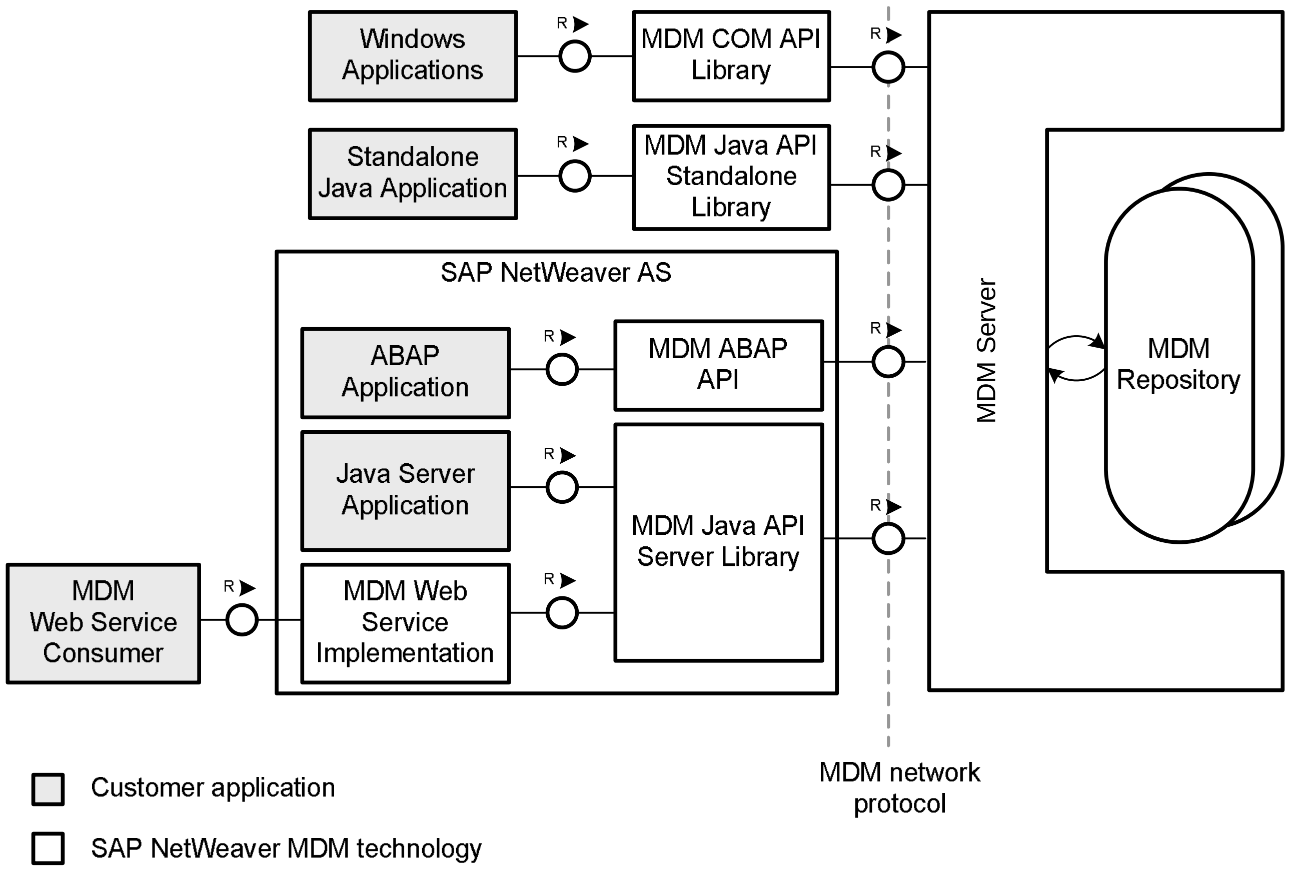


Figure 12 MDM APIs and Services

5.2.2 SRM Catalog Integration

The SRM-MDM catalog scenario is delivered as part of SAP SRM. With the SRM-MDM catalog scenario the procurement catalog in SAP SRM is populated with data from a SAP NetWeaver MDM repository. SAP SRM uses SAP’s open catalog interface (OCI) to access the SRM-MDM catalog. The SRM-MDM catalog uses the

MDM API to get the catalog content from the MDM server. See [10] for more information about the SRM-MDM catalog.

5.2.3 MDM Searching Capabilities

The search functionality of the MDM server offers two types of searches:

The *incremental drill down* search can be used to browse the content of the repository along multiple dimensions. In the MDM repository, the allowed values for fields of master data objects can be defined by using lookup tables. As described in detail in 5.3 lookup tables list the allowed values of fields in other tables. The incremental drilldown search uses the different lookup fields as the dimensions along which the search can be narrowed down by selecting one of the defined lookup values. Drill down searching is done by incrementally selecting one or multiple values for each dimension (i.e. lookup field). This way the result set is narrowed down step by step. When displaying the choices for a dimension, only those values are offered for which there are hits in the currently defined result set. This way the user never ends up with an empty result set.

The *free form search* allows searching for records by specifying search criteria for one or multiple fields. In addition, search expressions can be defined based on record fields using a variety of operators. With the free form search arbitrary search criteria are allowed, so it is possible that no hits are found.

5.2.4 Users and Authorization

The MDM server comes with its own user management and authorization concepts. The users, roles, and the assignment of users to roles are defined in the MDM repository. This means that users and roles are locally defined for each repository. The definition of a role lists the functions that can be executed by the members of this role (such as creating tables, running imports etc) and the access restrictions for tables and fields. For each field of each repository table, the access mode can be defined as read-only or read/write. An option for denying read access to specific fields or tables is planned. When access to a tables is defined for a specific role additional constraints can be defined. Constraints restrict access to only a part of the records in an MDM table. Constraints can be defined for lookup tables (5.3), for example to restrict access only to only a specific sub tree in a hierarchy. Constraints can also be used to restrict access to a specific repository mask (freely defined collection of master data objects, see 5.3.7). This way constraints can be used to model instance based authorization.

The MDM server can be configured to use an LDAP server for storing users and role assignments and for authenticating passwords.

5.2.5 Change Tracking

Using the MDM console, the MDM server can be configured to log changes made to the content of an MDM repository (adding, deleting, modifying records). For each repository the administrator can configure on field level which types of operations should be logged. A user interface for viewing change information is available in the portal and as a plug-in for the MDM data manager. The change tracking information covers not only content but also schema changes. For change tracking of meta data see also 5.3.9.

In the MDM console a log is available for each repository. The log in the MDM Console only lists the type and time of an event, for example at which time imports have been done into the repository. The detailed logs about specific import and export operations can be found in the file system of the MDM server (in the port folders, see 5.4.1).

5.2.6 Locking and Versioning

5.2.6.1 Basic Locking

The MDM server provides two mechanism for locking records that are being modified. The basic locking mechanism sets a write lock whenever a user starts to change the attributes of a record. If a second user tries to modify the same record concurrently, a notification is displayed that this record is already locked. The second user now has the option to either cancel the operation or seize the lock and thus to abort the changes of the first user. If the second user seizes the lock, the first user is notified that the operation was aborted by some other user and the changes done so far are discarded. The MDM Data Manager automatically commits changes applied to records when the input focus is moved to some other cell in the user interface (The current change can be cancelled by pressing the "escape" key). The locks are only active while one attribute of a data record is being changed. For a more powerful locking with full user control of the committing and rolling back of changes the check-out mechanism is provided.

5.2.6.2 Check Out Mechanism

In addition to the basic locking there is a cooperative check-out mechanism. This can be used optionally if basic locking is not sufficient. Before changing a data record, the user can choose to check out the record. This creates a private, temporary copy of the record. After the change is done, the user can either check in the changes or perform a rollback. In case of a rollback the temporary version is deleted and the changes are discarded. Checking in means that the original record is replaced with the temporary version. In both cases, the temporary version only exists while the record is checked out. Persisted versions that are kept as history after checking in are not supported.

When a record is checked out, it is locked for modifications by other users. However, the user who did the check-out can allow other users to join. If other users join a check out, the temporary version becomes available and writable for them as well. Records can be checked out in exclusive or non exclusive mode: in exclusive mode by default no one is allowed to join the checkout. The owner of the check-out has to explicitly allow other users or roles to join. In non-exclusive mode all users are by default allowed to join the checkout.

Check-out and check-in operations are also available via API and can be used to simulate transactions..

5.2.7 Platforms and Integration With Other SAP Technologies

The MDM server is available for various Unix platforms and Windows. For MDM repositories all relevant database platforms are supported. The GUI clients are available for Windows only. For other platforms the CLIX command line tools is available for server administration. Components like MDM Portal UI, MDM ABAP API and MDM enrichment architecture are available on the platforms supported by the SAP NetWeaver Application Server. See the platform availability matrix for details[4].

As stated above, the MDM server is a standalone application which is not running inside the SAP NetWeaver Application Server. This has the benefit of a fast installation and configuration without the need to configure a complete application server. On the other hand it means for customers that processes such as administration, error investigation, upgrade, and patching are different compared to other SAP applications. For SAP it means that infrastructure services of the application server like for example database abstraction, installation and update, user management, and security must be developed, maintained, and documented independently.

As explained above there is a rich set of interfaces for functional integration with SAP applications

- content level integration with SAP applications (see 5.1.1), via pre-defined extractors, schemas and mappings
- SAP SRM catalog integration (see 5.2.2)
- APIs and Web services (see 5.2.1)

Table 4 summarizes all other SAP NetWeaver components that can be part of a SAP NetWeaver MDM based solution. Note that these components are optional depending on the scenario.

[4] For up to date information about supported platforms see the platform availability matrix on the SAP service marketplace at http://service.sap.com/pam.

SAP NetWeaver Component	Role
SAP NetWeaver Portal	The SAP NetWeaver Portal is needed to run the MDM Portal UI (MDM iViews). Integration of MDM workflow items into the universal work list in the portal is also supported.
SAP NetWeaver Exchange Infrastructure (XI)	Used to import data from and distribute data to SAP application systems, SAP NetWeaver BI or third party systems (see 5.4)
System Landscape Directory (SLD)	SLD integration is used to add the MDM components to the list of installed software components in SLD. SLD is not required for MDM functionality.
CCMS	The MDM sends real-time technical monitoring data to a central monitoring system using the CCMS architecture.
TREX	Used for searching in documents (PDF) that are stored in the MDM repository
SAP NetWeaver BI	SAP NetWeaver BI can be connected to an MDM repository via SAP NetWeaver XI (like SAP application systems). SAP NetWeaver BI is supported as a remote systems and corresponding content is delivered.
SAP NetWeaver Application Server	Applications running on the SAP NetWeaver Application Server can connect to SAP NetWeaver MDM using the MDM APIs for ABAP and Java. The SAP NetWeaver AS Java is also needed to run the enrichment controller and enrichment adapters of the MDM enrichment architecture (see 5.5).

Table 4 Integration with other NetWeaver Components

5.3 MDM Repository Data Model

The fact that the MDM repository is persisted in a relational database is completely hidden from the MDM user and the MDM schema designer. The MDM repository provides a business abstraction above the relational database level. With this abstraction it is easy to model multi-valued and collection-like attributes as well as hierarchies and taxonomies. Based on these abstractions the MDM client provides a rich but generic user interface for browsing, searching and maintaining repository content.

5.3.1 Main Table and Sub Tables

The core entity of an MDM repository schema is the main table. Each MDM repository contains exactly one main table. The records of the main table represent business objects of a specific type (such as customer, employee or product). For each of these object types a separate MDM repository with a separate main table is required. This implies that relationships between those master object types cannot be modeled. This is a known limitation and the MDM development team is working on a solution that allows multiple main tables in the same repository.

As mentioned above, an MDM table is different from a relational table. A row in the MDM main table may have structured fields composed of multiple components and it may contain fields that are collections of structured records. This is achieved by using sub tables that describe complex fields in the main table.

The records of the main table can contain two types of fields: simple fields and lookup fields.

Simple fields have one of the following types:

- A simple type such as text (max 333 char Unicode), numeric (4 byte real), currency (8 byte real), date, time, or automatically generated identifiers, or
- A compound measurement type which is used for physical measurement values. When a measurement field is defined a physical dimension (such as current, pressure or weight) must be assigned. The field itself consists of the numeric value and the unit of measure (with allowed values depending on the physical dimension).

Lookup fields are fields that are defined based on sub tables that are used as lookup tables. There are different types of sub tables that are explained in the following sections:

- Flat sub tables,
- hierarchy tables,
- taxonomy tables, or
- qualified sub tables.

In addition there are predefined sub tables for content such as PDF documents, video and images. Sub tables can either be used as lookup tables or they can be associated with the main table using relationships (see 5.3.6.).

5.3.2 Flat Sub Tables

A flat sub table contains a set of values. It can be used to lookup the allowed values for the field in the referencing table (which may be the main table or another lookup table). In other words: The sub table defines the values of an enumeration type that is used as the type of a field in the referencing table.

5.3.3 Hierarchy Tables

Hierarchy tables are used when the allowed values of a lookup field are organized in a hierarchy. The hierarchy table defines a tree-like hierarchy of nodes that have identical structure. The definition of the hierarchy table lists the attributes of one node in the hierarchy. When the user fills the table using the MDM Data Manager, the table content is displayed as a hierarchy to which the user can add new nodes.

A field in a referencing table can be defined as a hierarchy lookup field that is based on some specific hierarchy table. In that case the allowed values for the field in the referencing table are the leaf nodes of the hierarchy tree described by the content of the hierarchy table.

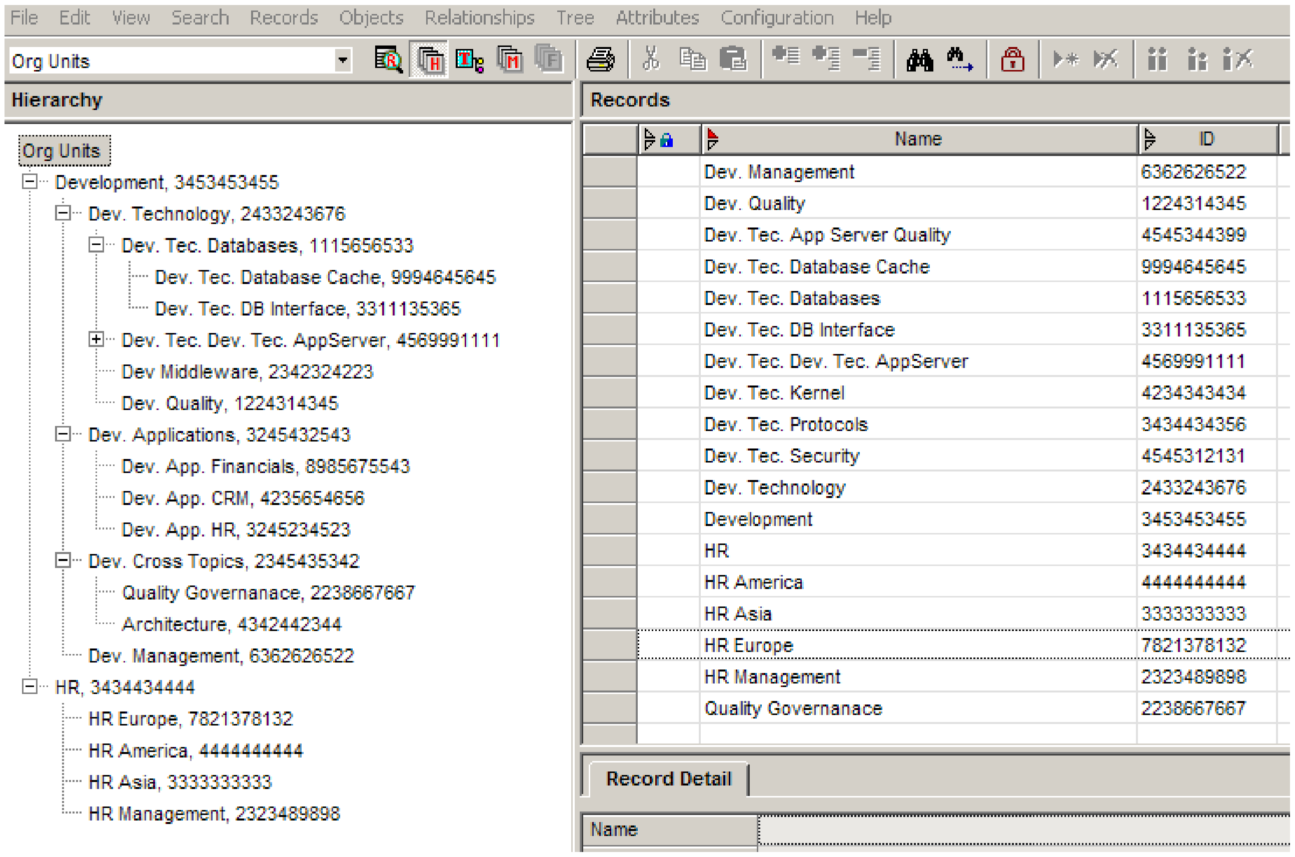
Example:

An example for using a hierarchy table is a repository for employees. Employees belong to organizational units which are the leaf nodes in a organizational hierarchy. A simplified data model for this example consists of a main table for employees and a hierarchy table for the organizational units. In the example the hierarchy table is named “OrgUnit” and contains two fields for name and ID. No additional attributes have to be defined for implementing the hierarchy. This is done automatically by the implementation of the MDM repository in a way that is hidden for the user. The main table for employees contains a field “org unit” that is defined as a lookup field based on the “OrgUnit” hierarchy table. Figure 13 shows how the hierarchy of organizational units is maintained in the MDM Data Manager[5]. After the nodes of the hierarchy have been defined, the user can switch to the main employee table and maintain the employee records. When the “org unit” field of an employee is maintained, the leaf nodes of the hierarchy in Figure 13 are offered as allowed values.

It is important not to misunderstand the name “hierarchy table”. Hierarchy tables are used to define allowed values for main table fields with the additional feature that these values themselves have a hierarchical

[5] The examples show test data based on a test schema. The test schema was only created to understand the concepts of the MDM product and it is much simpler than a real master data schema would be.

structure. The hierarchy relationship exists among the records of the hierarchy table. Hierarchy tables are not used to define hierarchies of nested objects with different types (such as buildings that contain floors that contain rooms that contain desks).



```
Org Units
⊟ Development, 3453453455
    ⊟ Dev. Technology, 2433243676
        ⊟ Dev. Tec. Databases, 1115656533
            Dev. Tec. Database Cache, 9994645645
            Dev. Tec. DB Interface, 3311135365
        ⊞ Dev. Tec. Dev. Tec. AppServer, 4569991111
        Dev Middleware, 2342324223
        Dev. Quality, 1224314345
    ⊟ Dev. Applications, 3245432543
        Dev. App. Financials, 8985675543
        Dev. App. CRM, 4235654656
        Dev. App. HR, 3245234523
    ⊟ Dev. Cross Topics, 2345435342
        Quality Governanace, 2238667667
        Architecture, 4342442344
    Dev. Management, 6362626522
⊟ HR, 3434434444
    HR Europe, 7821378132
    HR America, 4444444444
    HR Asia, 3333333333
    HR Management, 2323489898
```

Name	ID
Dev. Management	6362626522
Dev. Quality	1224314345
Dev. Tec. App Server Quality	4545344399
Dev. Tec. Database Cache	9994645645
Dev. Tec. Databases	1115656533
Dev. Tec. DB Interface	3311135365
Dev. Tec. Dev. Tec. AppServer	4569991111
Dev. Tec. Kernel	4234343434
Dev. Tec. Protocols	3434434356
Dev. Tec. Security	4545312131
Dev. Technology	2433243676
Development	3453453455
HR	3434434444
HR America	4444444444
HR Asia	3333333333
HR Europe	7821378132
HR Management	2323489898
Quality Governanace	2238667667

Figure 13 Hierarchy Table Example

5.3.4 Taxonomy Tables

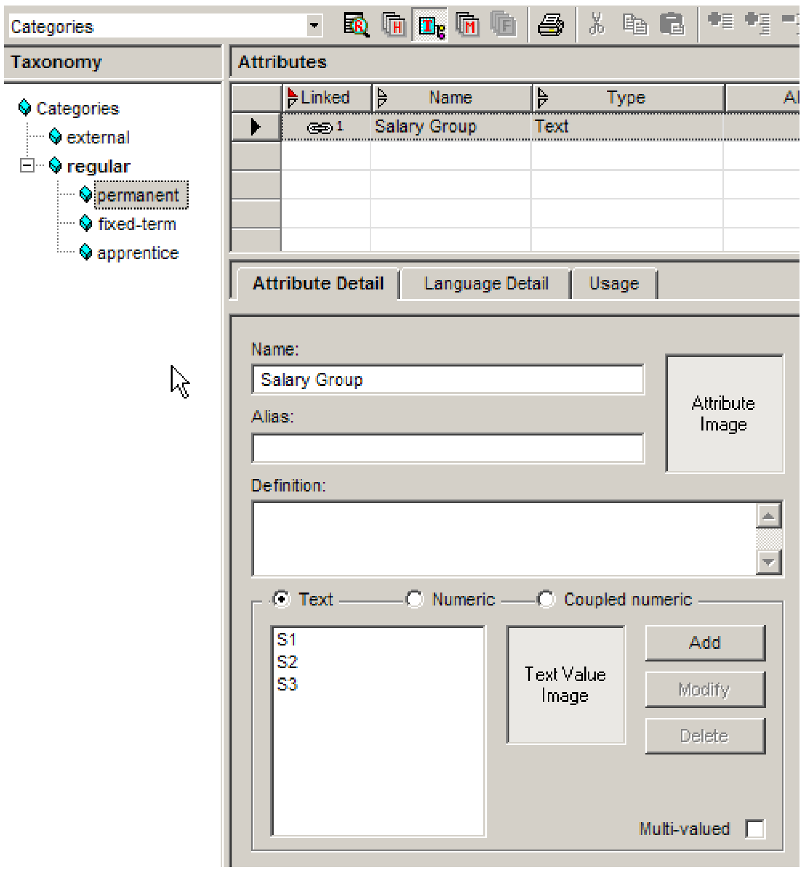
Taxonomy tables are similar to hierarchy tables but with slightly different semantics and additional features. A taxonomy table is used to model a classification hierarchy. Each record in the taxonomy table corresponds to one node in the classification hierarchy (i.e. one category in the taxonomy). Like with hierarchy tables the hierarchical relationship need not be modeled by the user but is defined implicitly. The allowed values for a taxonomy lookup field in the main table are the leaf nodes of the taxonomy hierarchy. Taxonomy tables can only be used to define lookup fields in the main table. Lookup tables cannot contain taxonomy lookup fields.

Compared to hierarchy tables, the taxonomy tables add the possibility to define attributes for the members of a specific category in the taxonomy. This feature can be roughly compared to the definition of fields for a class in object oriented programming. If an attribute is defined for a node in an MDM taxonomy, this attribute is also inherited by all nodes in the sub tree underneath that node.

Example

In Figure 14 we see an example for a taxonomy of personnel. The taxonomy is called “Categories”. This taxonomy table is used as a lookup table in the definition of the “category” field in the main employee table. Our simple example assumes that there is external personnel (contractors, freelancers, guests) and regular personnel. The regular employees are further categorized based on whether they have a permanent contract, a fixed term contract or whether they are apprentices[6]. An attribute “Salary Group” is defined and linked to the “regular” category. This means that external personnel does not have a salary group but all types of regular employees have such an attribute. As a consequence this attribute appears for all employee records in the employee main table that have the category field set to “permanent”, “fixed-term” and “apprentice”.

In the user interface in the MDM Data Manager, the taxonomy attributes appear like additional fields of a main table record. The difference to normal fields is that they are not defined in the static schema. Instead the attributes are defined and assigned to taxonomy categories when the taxonomy table is filled with values. This is reflected in the choice of tools that must be used: Ordinary fields are defined using the MDM Console, but taxonomy attributes are defined while creating content in the MDM data manager. The taxonomy attributes are currently limited to attributes that have an enumerated type (like salary groups S1,S2,S3 in the example) or a numeric type



Linked	Name	Type	Al
1	Salary Group	Text	

Name: Salary Group

Alias:

Definition:

[✓ Text] Numeric Coupled numeric

S1
S2
S3

Multi-valued ☐

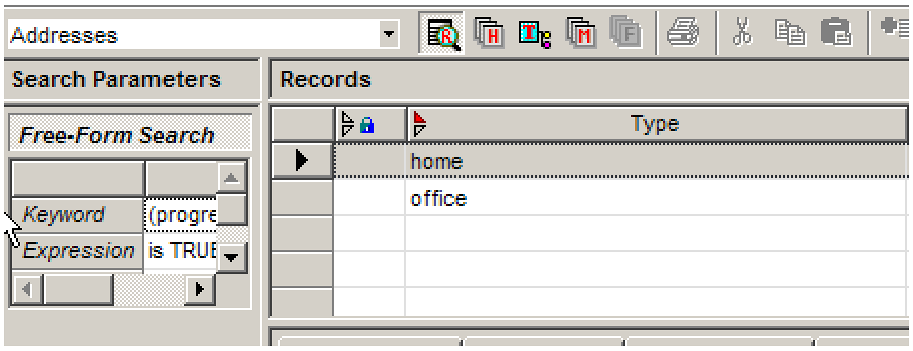
Figure 14 Taxonomy Table Example

5.3.5 Qualified Sub Tables

Qualified sub tables can be used to define main table fields that contain a collection of structured fields. The definition of a qualified sub table defines fields that are either qualifiers or non-qualifiers. The non-qualifier fields are like fields of an enumeration type. The allowed values for the non-qualifiers are defined when the qualified sub table is filled with content. For qualifier fields, the sub table declares only the existence and the types of the fields. The values of qualifier fields are not stored in the sub table. Qualifiers are rather attributes

[6] The reader should not think too much about the business logic of the example. It is only to show the modeling concepts and does not try be realistic at all.

of the association between a main table record and an associated non-qualifier in the sub table. Qualifier fields are instantiated and filled with values whenever a non-qualifier is assigned to a main table record. Another way to look at qualified tables is to describe the non-qualifiers as selectors and the qualifier fields as the values that are valid for a specific selector.



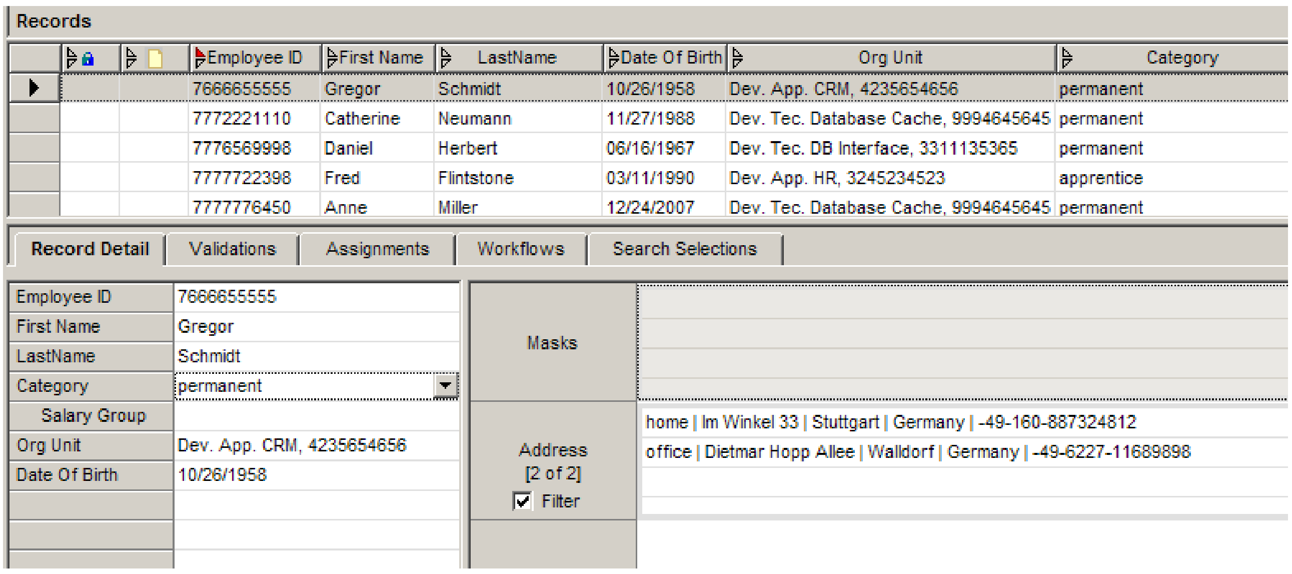
Records

Type
home
office

Figure 15 Example Content of a Qualified Lookup Table

Example

An example application of qualified lookup tables are addresses of employees: Assume that we have an employee main table that contains a lookup field “address”. The “address” lookup field points to a qualified sub table “Addresses”. The definition of the “Addresses” table lists one qualifier field “type” and the non-qualifier fields “country”, ”city”, ”street” and “phoneNumber”. When the Addresses table is filled with content, only the “type” field (the non-qualifier) field is filled with data. We assume that two types of addresses are maintained: “office” and “home” (see Figure 15). The qualifier fields are not maintained in the addresses table. They are shown and maintained in the “address” field of the individual records in the employee main table. In the example in Figure 16 two addresses are shown for the selected employee in the main table view of the MDM data manager. From there the address qualifiers can be maintained by double clicking the address details area.


Records

Employee ID	First Name	LastName	Date Of Birth	Org Unit	Category
7666655555	Gregor	Schmidt	10/26/1958	Dev. App. CRM, 4235654656	permanent
7772221110	Catherine	Neumann	11/27/1988	Dev. Tec. Database Cache, 9994645645	permanent
7776569998	Daniel	Herbert	06/16/1967	Dev. Tec. DB Interface, 3311135365	permanent
7777722398	Fred	Flintstone	03/11/1990	Dev. App. HR, 3245234523	apprentice
7777776450	Anne	Miller	12/24/2007	Dev. Tec. Database Cache, 9994645645	permanent

Employee ID	7666655555
First Name	Gregor
LastName	Schmidt
Category	permanent
Salary Group	
Org Unit	Dev. App. CRM, 4235654656
Date Of Birth	10/26/1958

Masks

Address [2 of 2] ✓ Filter

home | Im Winkel 33 | Stuttgart | Germany | -49-160-887324812
office | Dietmar Hopp Allee | Walldorf | Germany | -49-6227-11689898

Figure 16 Qualified Lookup Fields In a Main Table

The qualified table concept and the corresponding user interface offer a convenient way for modeling and maintaining nested data records. Unfortunately this feature is currently limited to a two-level nesting: Only the main table may contain qualified lookup fields. A qualified lookup table itself cannot contain qualified lookup fields. Without this limitation nested structures of arbitrary depth could be modeled.

5.3.6 Relationships

The MDM repository model allows the definition of an arbitrary number of relationships. With the MDM data modeling tools it is sufficient to declare the relationship. How the relationship is actually stored in the underlying implementation is hidden from the user.

The data modeling user interface and the documentation describe two possible types of relationships in an MDM repository: “sibling relationship” and “parent/child” relationships. Sibling relationships are symmetric relationships on the set of main table records. They can be used to model associations like for example the exchangeability of products. Symmetric means in this context that whenever the user creates a new relationship between records A and B, the MDM system automatically adds the corresponding relationship between B and A.

“Parent/child” relationships can be defined between main table and main table, the main table and a sub table or between sub tables. Despite the name (which may imply a hierarchy) this type of relationship can be used to model arbitrary n:m relationships.

The designer of an MDM schema has the choice to implement associations between main table records and sub table records in two different ways: It can be done with lookup fields in the main table or by using relationships to sub tables that are not used as lookup tables. The differences between the two options are

- Only lookup tables can be used for the iterative drilldown search (see 5.2.3). To allow efficient searching, lookup tables are intended to have not too many records.
- With lookup tables of type taxonomy and qualified sub table, the nesting is limited to one level of nesting. With relationships this limitation does not exist.
- Relationships cannot be exported using the syndication tools described in 5.4. Therefore relationships cannot be synchronized with the application systems in the usual and automated way. With this limitation, relationships are not useful for master data synchronization tasks.

Relationships can be used for one-way synchronization (import only) scenarios that use MDM APIs to retrieve the consolidated data from the repository. Relationships can also be used to do additional modeling in the MDM server that enriches data with new aspects that are not existing in the application systems. One example is the product catalog scenario, where the MDM server is used for consolidating and enriching product data which is then published directly from the MDM repository.

5.3.7 Repository Masks

MDM users can define an arbitrary number of named sets of master data records called masks. Masks can be used as an additional mechanism for defining sets of records, in addition to hierarchies and taxonomies. The difference is that masks are freely defined arbitrary collections. Masks are not defined based on rules or attributes but they are populated by interactively adding a selection of records or a search results to the mask. Masks can be used during incremental drill down search as another means to narrow down result sets. When working interactively on the MDM repository, masks can be used to mark a set of records that is meaningful for the users (for example records that the user still has to work on etc). Masks are stored in a special MDM system table named the Masks table.

5.3.8 MDM Meta Model

Figure 17 shows a simplified meta model of the SAP NetWeaver MDM repository. The meta model summarizes the content of section 5.3 and may serve as a reference. The model covers only the part of the repository model that is required for describing master data.

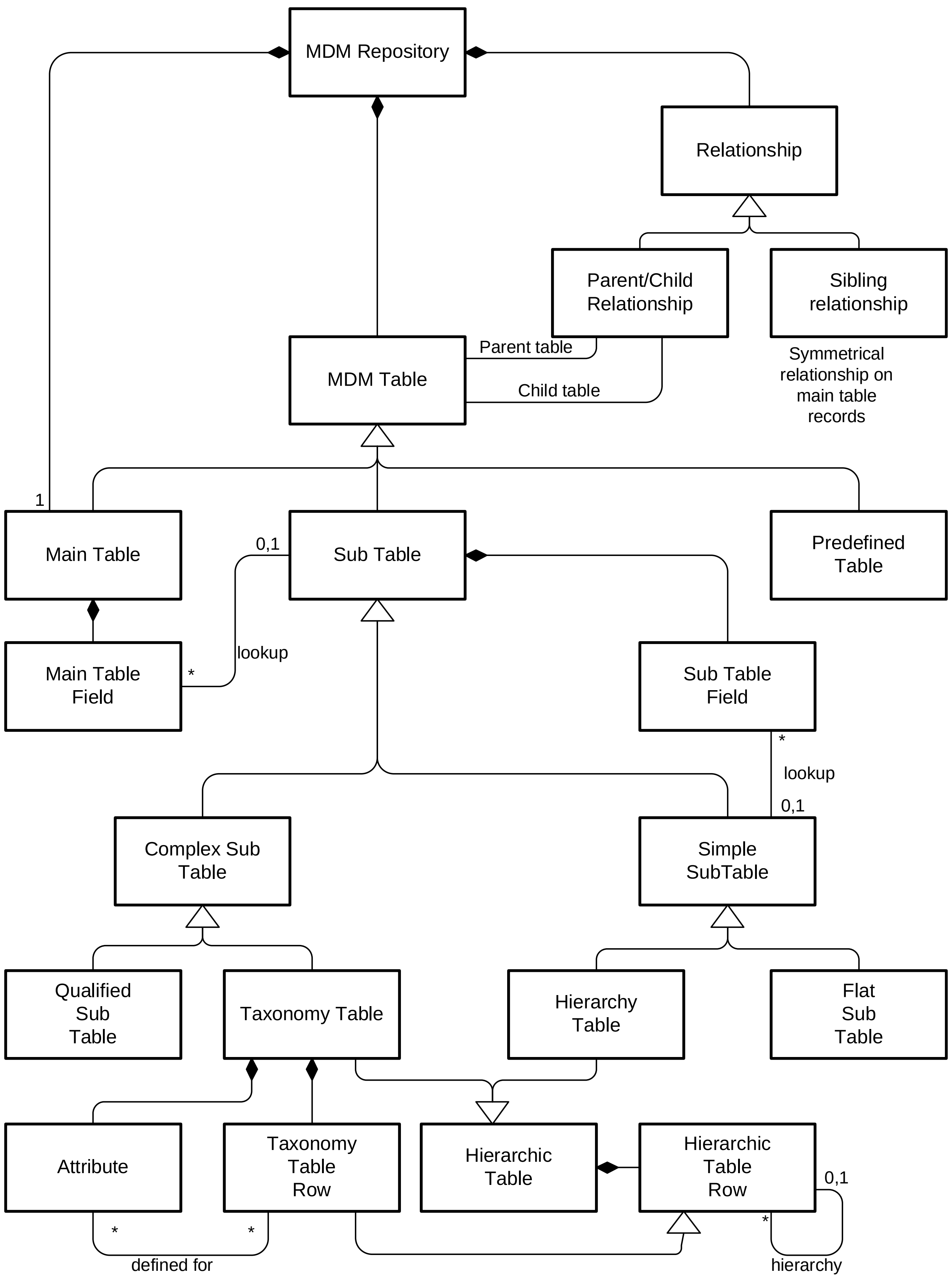


Figure 17 Simplified Meta Model of the SAP NetWeaver MDM Repository

The meta model shows that MDM repositories consist of tables which belong to one of three types

- Main table (only one per repository)
- Sub tables
- Predefined Tables. These are infrastructure tables which are automatically created for each new MDM repository. Many of these tables are not used for storing master data but they are used by the MDM server to store information about things like workflows, validations, roles, users or remote systems. As these tables are part of the server implementation and not part of the user defined schema, they are not covered in more detail in Figure 17. In addition there are predefined tables for multimedia content, for PDF documents and a table for defining product families that share common multimedia attributes. These tables are also omitted in Figure 17.

Sub tables can be either simple or complex (this distinction was introduced in this document for modeling purposes). Complex sub tables are either qualified sub tables or taxonomy tables. In addition to fields – which are parts of every table - taxonomy tables can also contain attributes. As described in 5.3.4 attributes of a taxonomy table can be linked to categories or sub categories of the taxonomy and they are inherited by all sub categories in the hierarchy underneath. The simple sub tables are either flat sub tables or hierarchy tables. Both hierarchy tables and taxonomy tables are hierarchic tables. Hierarchic tables have a hierarchy relationship defined on their rows.

Fields in MDM tables can be defined as lookup fields that point to a sub table. As shown in Figure 17, fields in the main table may point to any type of sub table but fields in sub tables may only point to simple lookup tables. Because the use of lookup tables is optional, the “lookup” associations in the model have many to zero or one cardinality. If a field has no lookup table, it is a simple field of a built-in type.

MDM repositories support relationships that are either sibling relationships or parent/child relationships. Sibling relationships are symmetric relationships on the set of main table records. Parent/child relationships are relationships between two tables (main/main, main/sub table or sub table/sub table).

5.3.9 Schema Change Management

In general development environments support change management for schema definitions and other meta data (like roles, remote systems, ports) to server two purposes:

- Change tracking: Who changed repository definitions, which elements and when?
- Change propagation: Propagate changes between a test repository and a productive repository or between a master repository and its slave repositories.

SAP NetWeaver MDM supports tracking of schema changes. All schema changes are logged in a database table. Basic change propagation is also supported: The MDM console allows to export the description of the current structure of an MDM repository to an XML file. The same XML file can be imported and merged into a target repository. During import differences in the structures are displayed and each difference can be accepted or rejected by the administrator.

Advanced change propagation features like for example collecting changes in change requests, definition of propagation paths, queues and schedules are currently not available. An integration with the change and transport system of the SAP NetWeaver Application server is also missing.

5.3.10 Sharing Tables between Repositories

Sharing data between repository means that the same table is visible with its content in multiple repositories. An example would be a table with the definition of country codes that is needed in different repositories. This is currently not supported by the product but it is a feature planned for the future.

5.3.11 Reusing Schema Definitions Between Repositories

Schema reuse means to use the same table definition (for example of a sub table) in multiple repositories. SAP NetWeaver MDM does not support global definitions that can be references from multiple repositories but reuse can be achieved by the schema propagation mechanism described above. The schema is exported from one repository and only the desired definitions are imported into the other repositories.

5.3.12 Multilingual Support

The MDM repository supports multiple languages on three levels:

- o Repository metadata like names for tables and fields can be maintained in multiple languages
- o The repository data can be entered in multiple languages.
- o The language of the MDM GUI clients can be switched using the MDM Language Selector utility.

For each MDM repository a list of repository languages can be defined. This is done by an administrator in the MDM Console application. The repository languages can be selected from the list of all languages that are supported by SAP NetWeaver MDM. Multilingual support for repository data is available for text fields and for binary fields like multimedia content. While multimedia fields are always multilingual, text fields must explicitly be defined as multilingual in order to get multi language support for the data values.

Records with fields that are configured as multilingual still appear as one record in the MDM Data Manager. The field values are shown in the language that was selected by the user during logon. In addition the language specific values of multilingual fields appear on the language detail tab of the selected record. There the translation of all multilingual fields can be maintained for every supported language.

SAP NetWeaver MDM supports a feature called language inheritance: If a translation is missing in the required language, the translation is taken from the next language in an ordered list of fallback languages. Two disjoint lists of fallback languages can be defined for each language: The primary language inheritance list defines fallback languages that are good enough to be used in publications (for example in a product catalog). For example: A useful primary inheritance substitute for English [US] is English [UK]. The secondary language inheritance list defines the fallback languages that are not acceptable for publishing but which are useful for displaying in MDM clients. In the UI of the MDM clients a color code is used to indicate whether a value is from the current language, from a language in the primary inheritance list or from a language in the secondary inheritance list. Language inheritance lists can be defined separately for each language.

5.4 Master Data Synchronization: Import and Distribution

5.4.1 The Port Concept

For defining the inbound and outbound channels to remote systems SAP NetWeaver MDM uses the concept of ports. This concept is explained using the topic map in Figure 18.

In each repository an unbounded number of remote systems can be defined. An example for a remote system could a be a client in an SAP system. For example, remote system “ERP100” could represent client 100 in system “ERP”. For each remote system multiple uniquely named ports can be defined. On the MDM server, a file system folder is created for each port. In this folder the server stores for example the communication log files. A property of the port is the type of communication (ODBC, XML, plain file etc). For type “XML” exactly one XML schema must be specified that describes the type of documents that are exchanged via the this port. For all file based types of communication the folder that corresponds to the port is also used as the inbox or outbox for messages that are sent to or received from the remote system.

A port is either an inbound port or an outbound port. For inbound ports, exactly one import map must be specified. The import map defines the structural mapping and value transformations for incoming data. In case of outbound ports a “syndication map” must be defined. The syndication map defines the mappings for exporting data from the MDM repository. For outbound ports it is also possible to define key ranges that restrict the values of remote keys that are generated for master data records that do not yet have a corresponding record in the remote system. It is possible to define either a normal range for the generated keys or a qualified key range. A qualified key range is defined based on the values maintained in a taxonomy or hierarchy lookup table (see 5.3). For each leaf node in the hierarchy a different (non-overlapping) key range can be defined. This can be used, for example, to define different ranges for the remote key of a product (product ID) for different products categories that are defined in a taxonomy table.

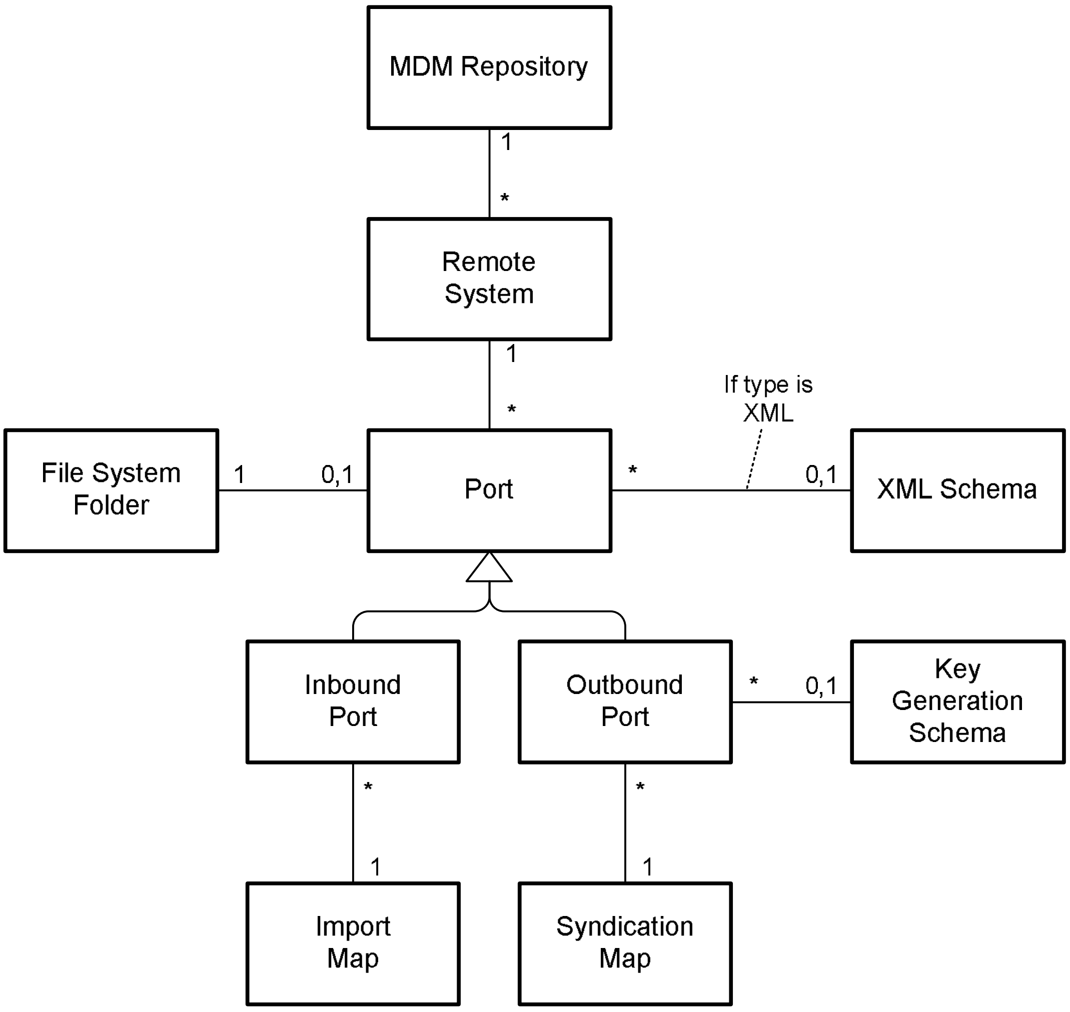


Figure 18 The MDM Port Concept

5.4.2 Importing Master Data Into The Repository

5.4.2.1 Import Architecture

Details of the import process are shown in Figure 19: The master data records are exported from an SAP ERP system as IDocs and accepted by the IDoc adapter of SAP NetWeaver XI. SAP NetWeaver XI transforms the IDoc into an XML message and places it into the inbox folder of an inbound port of the MDM server. The XI File/FTP adapter is used to deliver the file on the receiver side. The communication channel in SAP NetWeaver XI is configured to place the file into the inbox folder of an inbound port of the MDM server. During regular operation the MDM Import Server is used to poll the folders of the inbound ports and automatically import arriving data. The polling interval can be specified as a configuration parameter. The import server uses a previously created import map to transform the XML document into records that can be stored in the MDM repository. Finally the MDM Import Server updates the MDM repository with the received data.

Figure 19 also shows the MDM Import Manager. This is the application that is used to interactively define the import map that is later used by the MDM Import Server. With the MDM Import Manager the elements of the source schema are mapped to the tables and fields of the MDM repository. Different mappings can be defined for different elements/tables. Typically one mapping is defined for the main business object type (which is stored in the MDM repository's main table). For synchronizing the content of lookup tables there are two possibilities:

- The data in the sub table data are synchronized together with the main table content by adding the linked lookup values to each main table record. This requires only one XML schema but has the disadvantage of redundancy: complete lookup table records must be included with every main table record (instead of only adding an identifier).
- The content of the lookup tables can be synchronized separately with their own mappings and their own XML schema. Typically the lookup tables contain data that are changed less frequently than the content of the main table. For example: The country names and country codes or the organizational structure. The usual pattern is to synchronize the lookup table content initially, often as a manual step. Later these tables are synchronized only on demand, while the synchronization of main tables happens continuously and often in an automated way.

The MDM Import Manager can also be used to manually import master data. It can also be used to analyze and resolve problems that occurred during automated import. If such an exception occurs the MDM Import Server logs the problem and places the affected files into a specific sub folder of the inbound port folder. These files can be opened in the MDM Import Manager for fixing the problem and importing the records manually.

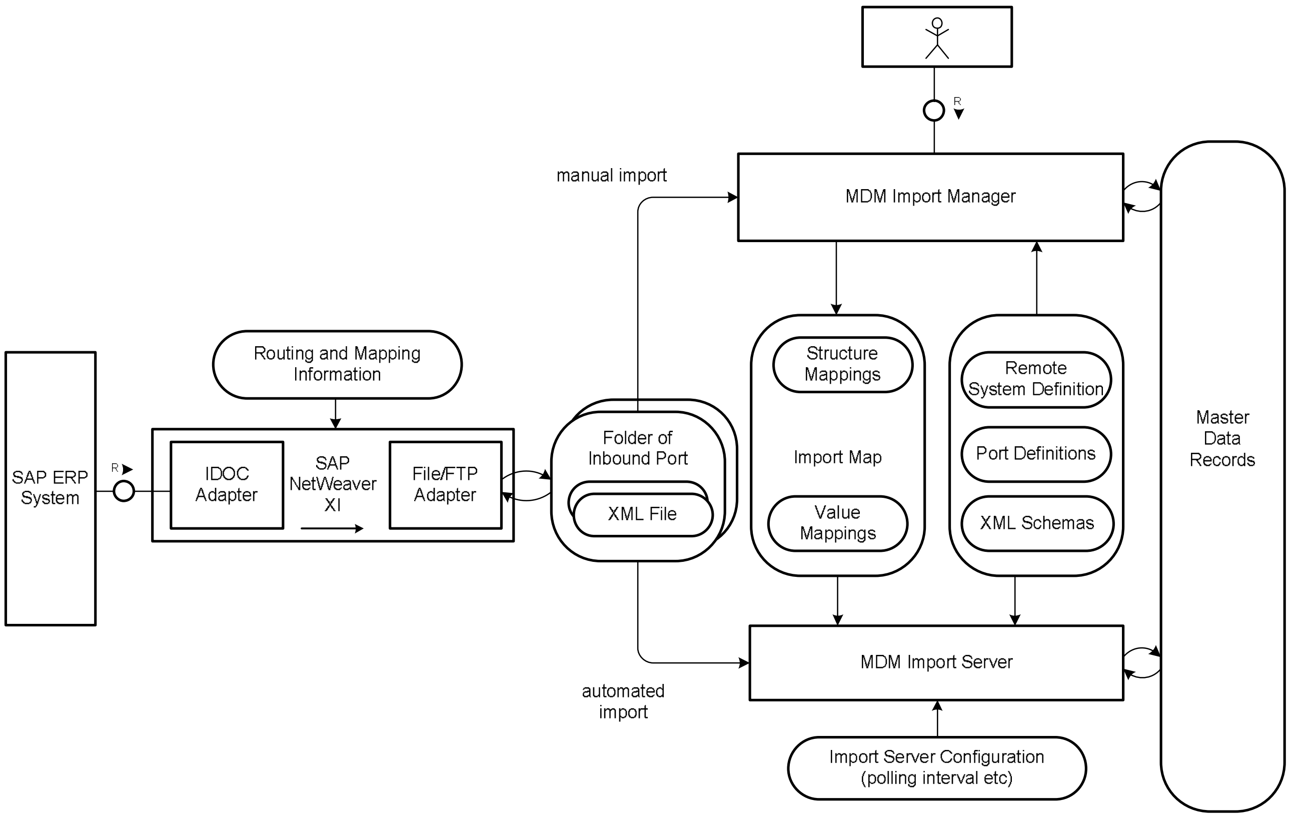


Figure 19 Importing Master Data

5.4.2.2 Staging

Master data management systems typically use some kind of staging during import. Inbound data is first imported into a staging area, where it can be validated and cleansing activities like completing, correcting and merging can be done. Only after an approval step the data is made visible for all users. SAP NetWeaver MDM supports staging in two steps:

During import the MDM Import server checks incoming data for technical correctness. If an error is detected the input file is placed in an exception folder of the inbound port and a problem is signaled in the MDM console. An expert has to correct the problem using the MDM Import Manager. This concept ensures that technically incorrect data are not imported into the repository

As a second step “virtual staging” can be used to ensure that the imported data records are hidden from the users until they are cleansed and approved. Virtual staging is achieved by importing the records into the MDM repository in checked out state (this is done by setting an option in the import map). The checked out versions are not visible except for the user who checked out the version or for users who joined the checkout (see 5.2.6.2). Typically an MDM workflow is defined for processing the imported data using virtual staging. In the import map it can be defined that a workflow is triggered on completion of the import operation. Typical steps of such a workflow are the interactive cleansing of the imported records, external enrichment like address checking (see 5.5), executing validations, merging and approval. The last step in the workflow is checking-in the approved records to make them visible for all users.

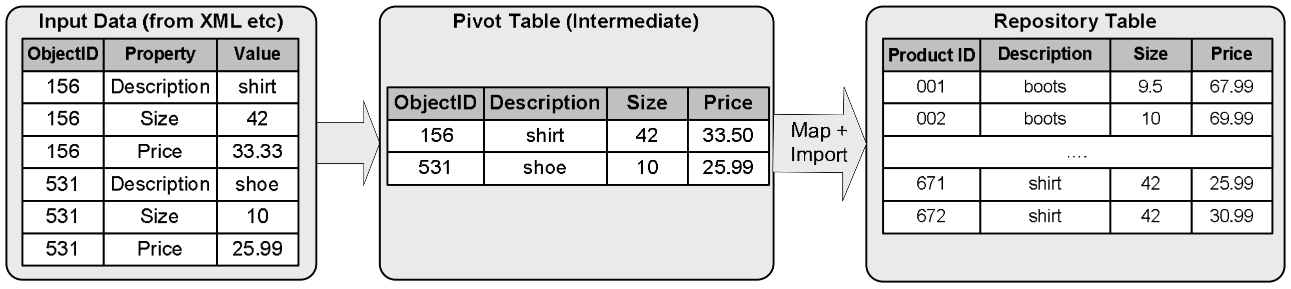
With the concept of virtual staging no new storage or processing environment is required. The full functionality of the MDM tools (like searching, matching etc) can be used already in the staging phase.

5.4.2.3 Import Maps

The definition of an import map may consist of the following steps:

Joining input tables/elements. Mapping in the MDM Import Manager is table based. The fields and values of an input table are mapped to the fields and values of an MDM repository table. Often the input structure consists of multiple tables or for example of multiple XML elements (which are treated as separate tables by the MDM Import Manager). In order to map multiple input tables (or a whole hierarchy of XML elements) to one target table, the input tables can be joined into an appropriate view. It is for example possible to virtually add fields of parent or child elements to an XML element before doing the mapping. The result of this step is a “virtual” input table which is used as the starting point for all subsequent steps.

Pivoting or reverse pivoting. Before fields and values are mapped the input data can optionally be transformed using pivoting or reverse pivoting. Pivoting combines multiple input records into one record of a resulting pivot table which has more fields. The values of selected input fields are turned into field names of the pivot table. A typical example is shown in Figure 20. The input table in the example contains an “ObjectID” and two fields that contain a property and the value of that property. The field named “property” can be viewed as meta data because it describes the purpose of the data in the “value” field. The pivoting operation combines all input records with the same “ObjectID” into one resulting record. The data in the “property” input field are used as field names taking the field values from the “value” field of the corresponding input record. Reverse pivoting distributes the fields of one input record to multiple output records turning the field names into values of “metadata” fields. The result of pivoting or reverse pivoting is an intermediate table which is used the actual input for the import mapping steps described below.



ObjectID	Property	Value
156	Description	shirt
156	Size	42
156	Price	33.33
531	Description	shoe
531	Size	10
531	Price	25.99

ObjectID	Description	Size	Price
156	shirt	42	33.50
531	shoe	10	25.99

Product ID	Description	Size	Price
001	boots	9.5	67.99
002	boots	10	69.99
….			
671	shirt	42	25.99
672	shirt	42	30.99

Figure 20 Pivoting During Import

Splitting and combining fields

Single fields of the original import records can be split into multiple fields that can be individually mapped to different target fields. It is for example possible to split a field consisting of multiple delimited parts into a hierarchy, into a taxonomy or into multiple independent fields. The reverse is also possible: Multiple input fields can be combined and the result can be mapped to a single target field.

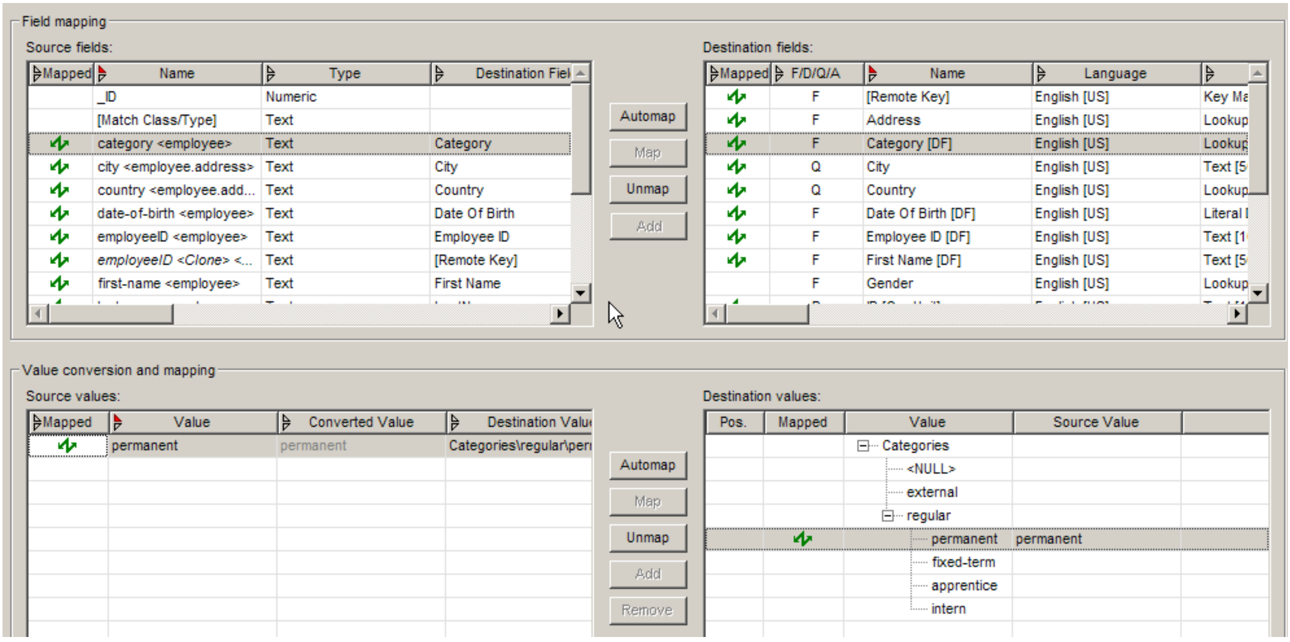
Value transformations.

Normalizations, transformations and data type conversions can be applied to input values before they are mapped and imported.

Structural mapping

Structural mapping defines which elements of the source schema map to which table fields in the target schema (see Figure 21).

When defining the structural mapping, source fields can also be mapped to the remote key in the MDM repository. For each repository table with key mapping enabled, a remote key can be stored for each remote system. By mapping a source field to the remote key, the value of that field is stored with the imported record and it can be used during export to identify the corresponding record in the remote system.


Field mapping

Source fields:

Mapped	Name	Type	Destination Fiel
	_ID	Numeric	
	[Match Class/Type]	Text	
✓	category <employee>	Text	Category
✓	city <employee.address>	Text	City
✓	country <employee.add...	Text	Country
✓	date-of-birth <employee>	Text	Date Of Birth
✓	employeeID <employee>	Text	Employee ID
✓	*employeeID <Clone> <...*	Text	[Remote Key]
✓	first-name <employee>	Text	First Name

Destination fields:

Mapped	F/D/Q/A	Name	Language	
✓	F	[Remote Key]	English [US]	Key Ma
✓	F	Address	English [US]	Lookup
✓	F	Category [DF]	English [US]	Lookup
✓	Q	City	English [US]	Text [5
✓	Q	Country	English [US]	Lookup
✓	F	Date Of Birth [DF]	English [US]	Literal I
✓	F	Employee ID [DF]	English [US]	Text [1
✓	F	First Name [DF]	English [US]	Text [5
	F	Gender	English [US]	Lookup

Value conversion and mapping

Source values:

Mapped	Value	Converted Value	Destination Valu
✓	permanent	permanent	Categories\regular\per

Destination values:

Pos.	Mapped	Value	Source Value	
		Categories		
		<NULL>		
		external		
		regular		
	✓	permanent	permanent	
		fixed-term		
		apprentice		
		intern		

Figure 21 Mapping Fields and Values For Import

Value mapping: This must be done for the content of lookup tables. For example: The employee categories used in the source system are mapped to the corresponding employee categories used in the MDM repository (see Figure 21).

Identity mapping: This defines which imported records should be mapped to which record in the repository. Based on this information the MDM Import Manager knows when to create a new record or when to update an existing record.

Defining the import action. Here a selection must be made between creating, updating and skipping records. It also possible to define the default behavior for non matching lookup table content: Ignore the new value, add the new value or replace it by a default value.

5.4.3 Master Data Distribution

5.4.3.1 Master Data Distribution Architecture

The block diagram in Figure 22 shows the components needed for distributing master data. The diagram assumes that definitions of remote systems and outbound ports were already created by an administrator. For each outbound port there is also an XML schema that describes the XML documents that are sent. These XML schemas are either delivered by SAP as business content or they are designed by the customers to fit their specific needs. Before the automated distribution can start, the mapping between repository content and the target XML structure must be defined. By mapping only selected attributes it can be defined which attributes of the objects in the MDM repository should be replicated to a specific remote system and which should be omitted. The mappings are defined interactively using the MDM Syndicator application. ("Syndication" is the term used by SAP NetWeaver MDM instead of "export" or "distribution"). The result of this process is the syndication map.

After the syndication map is defined and tested, it is associated to the definition of the outbound port. Then the automated syndication can start. The MDM Syndication Server checks periodically if there are new or changed records in the MDM repository that must be distributed. If new data are there, it uses the port definitions, XML schemas and syndication maps to create the required XML document for each outbound port. In SAP NetWeaver XI an inbound File/FTP adapter is configured to poll the outbox folder of the outbound ports. From there SAP NetWeaver XI takes the new XML document, transforms it into an IDoc and delivers it to the target SAP ERP system. Note that any type of target system and outbound message format can be used that are supported by SAP NetWeaver XI. SAP ERP and IDoc are just examples. As shown before (Figure 11) master data records can also be exported to different file types or SQL databases if a non-SAP system is connected without using SAP NetWeaver XI.

Connecting remote systems with a files and folders based interface provides high flexibility and a low entry barrier especially for integrating third party systems, legacy systems and third party middleware. One disadvantage of this loosely coupled architecture is the lack of end-to-end monitoring for data distribution. The MDM server does not get any feedback about whether distributed data really arrived in the target system. The MDM operator has to check in SAP NetWeaver XI and in the remote systems for MDM related errors manually.

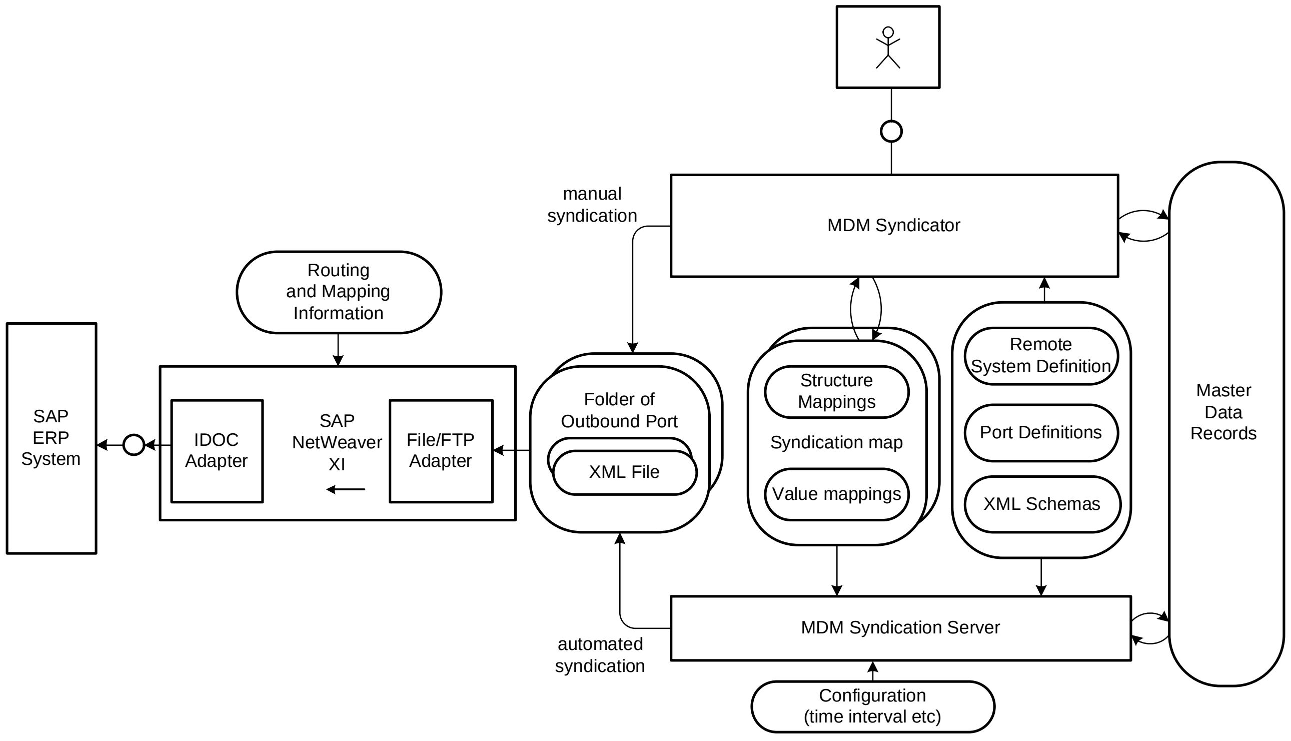


Figure 22 Exporting and Distributing Master Data

5.4.3.2 Outbound Filtering and Mapping

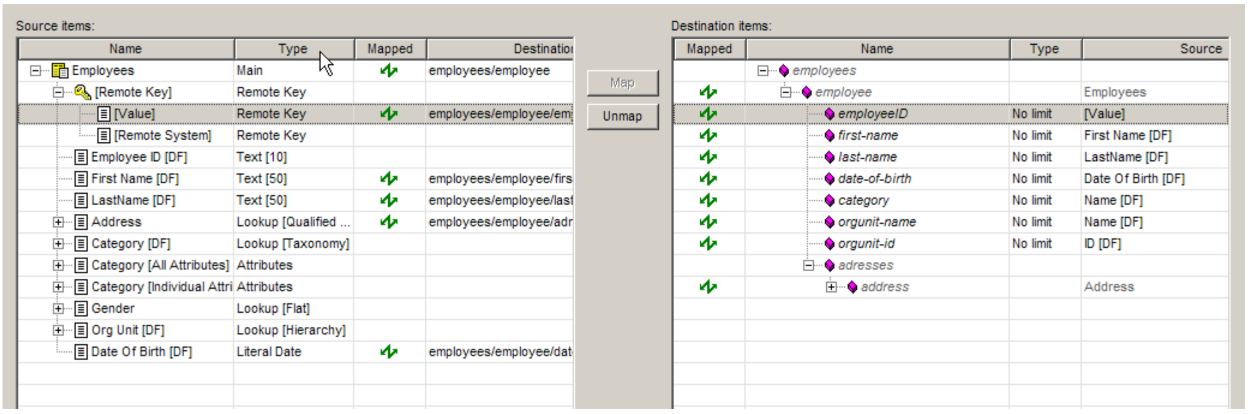
A syndication map defines which pieces of data are exported for distribution to a specific remote system and how the data in the MDM repository are mapped to the external format. The first step is filtering. Filtering defines which records are exported. The following filter criteria can be applied:

- Records can be filtered by search conditions based on the mechanisms described in 5.2.3. The search conditions are stored as part of the map and applied during automated export. This feature can be used for example to define that only employee data for a specific branch in the organizational hierarchy shall be distributed to a particular system.
- It is possible to configure that only those records are exported that were changed since the last export process. After the initial full distribution this switch is usually set and only deltas are exported. The MDM server only determines deltas between exports. Whether these exports actually reached the target systems is outside the scope of the MDM server.
- It is possible to define whether a remote system receives only updates of existing data or also new data.

After the filtering, mapping is applied to the results. The following mapping features are provided:

- Value normalization and formatting. Values can be transformed into the format required by the target system. In addition, default values can be defined for null valued fields.
- Combining fields. The values of multiple fields in the MDM repository can be merged together with predefined fixed texts into one virtual source field. The resulting merged field can then be mapped to a field in the target structure.
- Splitting fields. Multi valued source fields or multi-language source fields can be split into separate virtual fields which can then be mapped individually to elements of the target structure.
- Exporting hierarchical values from hierarchy or taxonomy tables. This can be done by flattening the hierarchical path into a string separated by delimiters or by splitting the hierarchy field into multiple "virtual" source fields that are mapped individually.
- Exporting qualified lookup fields. The fields of qualified lookup tables can be mapped like fields of the main table. If multiple values exist, they can be mapped to a sequence of sub elements in a target XML structure.
- Destination specific keys. When the syndication map is defined, the key mappings stored in the MDM repository can be used to ensure that the exported records contain the correct identifier that is valid in a remote systems (see Figure 23).

The screenshot in Figure 23 shows the definition of the syndication map for the simplified employee schema that was already used in previous sections.



Source items:

Name	Type	Mapped	Destination
Employees	Main	✓	employees/employee
[Remote Key]	Remote Key		
[Value]	Remote Key	✓	employees/employee/em
[Remote System]	Remote Key		
Employee ID [DF]	Text [10]		
First Name [DF]	Text [50]	✓	employees/employee/firs
LastName [DF]	Text [50]	✓	employees/employee/last
Address	Lookup [Qualified ...	✓	employees/employee/adr
Category [DF]	Lookup [Taxonomy]		
Category [All Attributes]	Attributes		
Category [Individual Attri	Attributes		
Gender	Lookup [Flat]		
Org Unit [DF]	Lookup [Hierarchy]		
Date Of Birth [DF]	Literal Date	✓	employees/employee/dat

Destination items:

Mapped	Name	Type	Source
	employees		
✓	employee		Employees
✓	employeeID	No limit	[Value]
✓	first-name	No limit	First Name [DF]
✓	last-name	No limit	LastName [DF]
✓	date-of-birth	No limit	Date Of Birth [DF]
✓	category	No limit	Name [DF]
✓	orgunit-name	No limit	Name [DF]
✓	orgunit-id	No limit	ID [DF]
	adresses		
✓	address		Address

Figure 23 Definition of a Syndication Map

5.4.4 Data Ownership And Routing

With the concepts of remote systems and ports and with the maps for import and export, the SAP NetWeaver MDM system allows to define from which systems data is imported and to which system data is distributed. For each system it can be defined on attribute level which data is imported or distributed respectively and data can be filtered based on values. It is also possible to define whether new objects are created during import or whether only updates are accepted. With this functionality data ownership rules can be implemented in a sense that the MDM repository receives new or changed master data only from specific remote systems. Data routing rules can be implemented by creating the appropriate outbound ports and syndication maps. However, the implementation of these rules are embedded in the definition of ports and mappings. There is no high level view that provides a quick answer to questions like “in which systems is this type of records/attributes maintained?” or “to which systems are these data distributed?”.

5.5 MDM Enrichment Architecture

The MDM enrichment architecture is used for integrating third-party data enrichment services with SAP NetWeaver MDM. SAP NetWeaver offers a certification program for data quality partners who implement external services for this architecture. A typical scenario for using external enrichments services is the completing and correcting of imported master data before the matching and merging of duplicates and identicals (see also 5.6.1). In addition to the enrichment of existing data the same architecture could also be used to get data from external business information providers into the MDM repository and then distribute it across the landscape.

Technically a data enrichment service can be viewed as a function that takes a collection of data records and returns a collection of enhanced records. The data enrichment services do not have access to the whole MDM repository so the context that is available for their operations is limited. The enrichment services typically perform operations that can be defined based on the context of a single record. Examples are address standardization, data normalization or lookups against a standard repository.

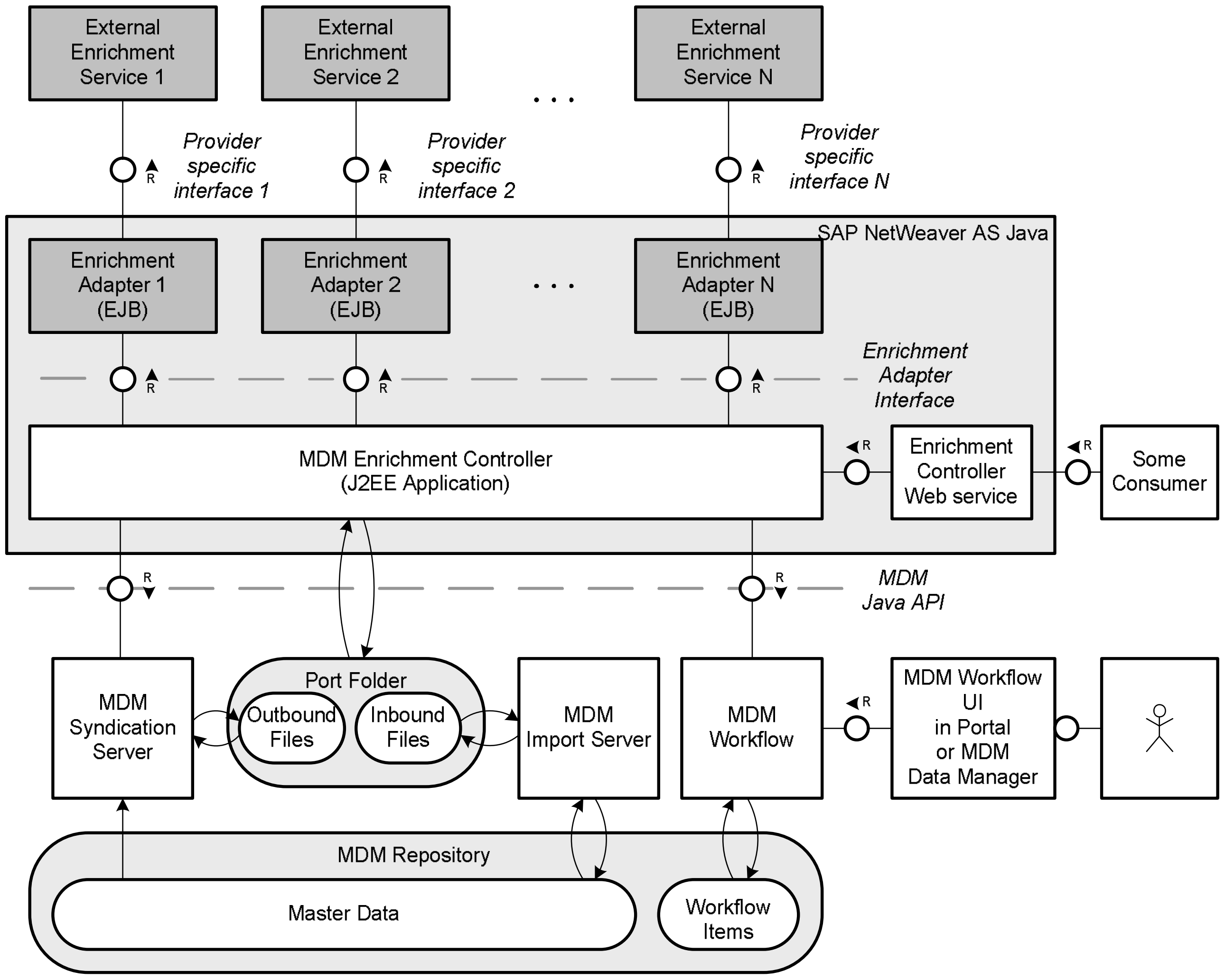

External Enrichment Service 1
External Enrichment Service 2
External Enrichment Service N
Provider specific interface 1
Provider specific interface 2
Provider specific interface N
SAP NetWeaver AS Java
Enrichment Adapter 1 (EJB)
Enrichment Adapter 2 (EJB)
Enrichment Adapter N (EJB)
Enrichment Adapter Interface
MDM Enrichment Controller (J2EE Application)
Enrichment Controller Web service
Some Consumer
MDM Java API
MDM Syndication Server
Port Folder
Outbound Files
Inbound Files
MDM Import Server
MDM Workflow
MDM Workflow UI in Portal or MDM Data Manager
MDM Repository
Master Data
Workflow Items
R

Figure 24 The Enrichment Architecture

The detailed MDM enrichment architecture is shown in the block diagram in Figure 24. Central component of this architecture is the enrichment controller, which is implemented as a J2EE application on the SAP NetWeaver Application Server. The enrichment controller gets master data records from the MDM repository and passes them to a service specific enrichment adapter that translates the request into a call of the external enrichment service. For each external enrichment service a separate enrichment adapter must be provided. Enrichment adapters are not delivered by SAP. They are developed by the companies that provide the external services or by independent software vendors. Technically the enrichment adapters are Java Enterprise Beans that implement an adapter interface defined by the enrichment architecture.

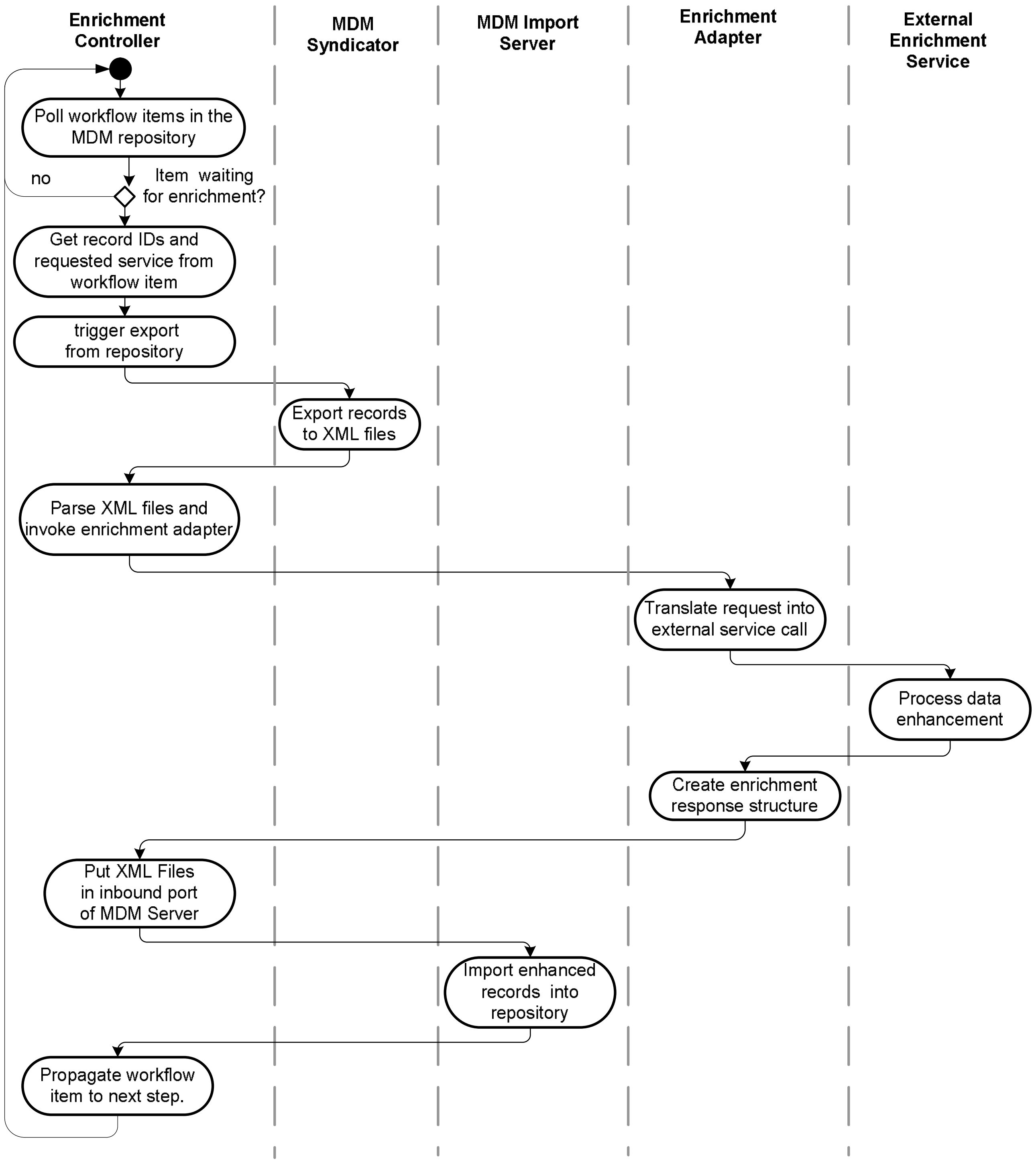

Enrichment Controller
MDM Syndicator
MDM Import Server
Enrichment Adapter
External Enrichment Service
Poll workflow items in the MDM repository
no
Item waiting for enrichment?
Get record IDs and requested service from workflow item
trigger export from repository
Export records to XML files
Parse XML files and invoke enrichment adapter
Translate request into external service call
Process data enhancement
Create enrichment response structure
Put XML Files in inbound port of MDM Server
Import enhanced records into repository
Propagate workflow item to next step.

Figure 25 Enrichment Architecture: Interaction of Components

The enrichment controller does not directly read from and write to the repository. Instead it uses the MDM Import Server and the MDM Syndication Server to export and import records from the repository. When new external enrichment services are added to an MDM installation, the MDM administrator has to define remote systems and ports for each external enrichment service. In addition the corresponding import and syndication maps must be set up.

Enrichment operations are normally executed as steps in an MDM workflow. Depending on the workflow definition the enrichment step can be triggered interactively from the MDM Data Manager or from the portal or it can be triggered automatically for example after an import. In addition to executing data enrichment as a part of an MDM workflow it is also possible to invoke the enrichment controller directly. This is done by calling the MDM Enrichment Web service.

The activity diagram in Figure 25 shows how the components of the enrichment architecture interact when an enrichment operation is executed as a step in an MDM workflow. The enrichment controller continuously polls the MDM repository for workflow items that are waiting for the execution of an enrichment step. If workflow items are found the enrichment controller retrieves the identifiers of the records from the workflow items and tells the MDM Syndication Server to export the records from the MDM repository. The enrichment controller reads the exported XML files, transforms the data into an enrichment request and calls the enrichment adapter that is responsible for the requested external service. The enrichment adapter translates the enrichment request into the required external format and invokes the external service. After the external service operation is finished the enrichment adapter translates the results into an enrichment response. The enrichment response with the enhanced master data is then returned to the enrichment controller. The enrichment controller transforms the results into an XML document and writes it to a file in the corresponding inbound port folder. From there it is picked up and processed by the MDM Import Server. After the import is done the enrichment controller updates the workflow item by marking the enrichment step as completed.

5.6 MDM Features For Ensuring Data Quality

5.6.1 Managing Duplicates And Identicals

In section 3.1.1 we defined duplicates and identicals as multiple occurrences of the same real life objects either in the same system or in multiple systems. After the records are imported into the MDM repository these two cases are technically treated in the same way. In both cases there are multiple records for which key mappings must be managed and attributes must be kept consistent. The first step is to find the records that correspond to the same real life entity. This step is called matching. After the sets of matching records are identified, the records are merged into a single record in the MDM repository.

5.6.1.1 Matching

The data flow for matching two records is depicted in Figure 26. During matching each record of a source set is matched against all records of a target set. Source and target set can be defined as all records in the repository, all selected records or the result set of a previous search. The diagram in Figure 26 explains the matching of two records: As a first step an optional normalization and transformation can be applied to the values of the fields of the records. The transformations are basically replacements of substrings or token by a substitution string. This can be used for example for expanding abbreviations or for replacing regional versions of a name with a standard version. (This step is optional. In the model in Figure 26 we assume that the values are simply copied if no transformation is configured.)

In the next step the matching rules are applied. Each matching rule consists of a condition and a set of scores. The condition contains a list of fields that must be equal, either exactly or based on a token match (percentage of matching tokens in the two strings). In addition, three scores must be defined: The score for the case that the values match, the score for the case that the values do not match and the score for the case that at least one of the values compared is not defined (null). The output of the matching step is one score for each rule which describes how good the match was for that rule. By assigning different scores to the rules, the rules can be weighted compared to the other rules. In the final step the matching strategy is applied. The matching strategy defines a set of rules as input, the scores of which are added to get a total score. This final score is then assigned to one of the three result categories “no match”, “low match” and “high match”. Different matching strategies can be defined and stored in the repository with different sets of input rules and different thresholds for the final classification. For executing the matching the source and target sets and the matching strategy have to be specified. Matching cane be done interactively, via API or automated as a step in a MDM workflow.

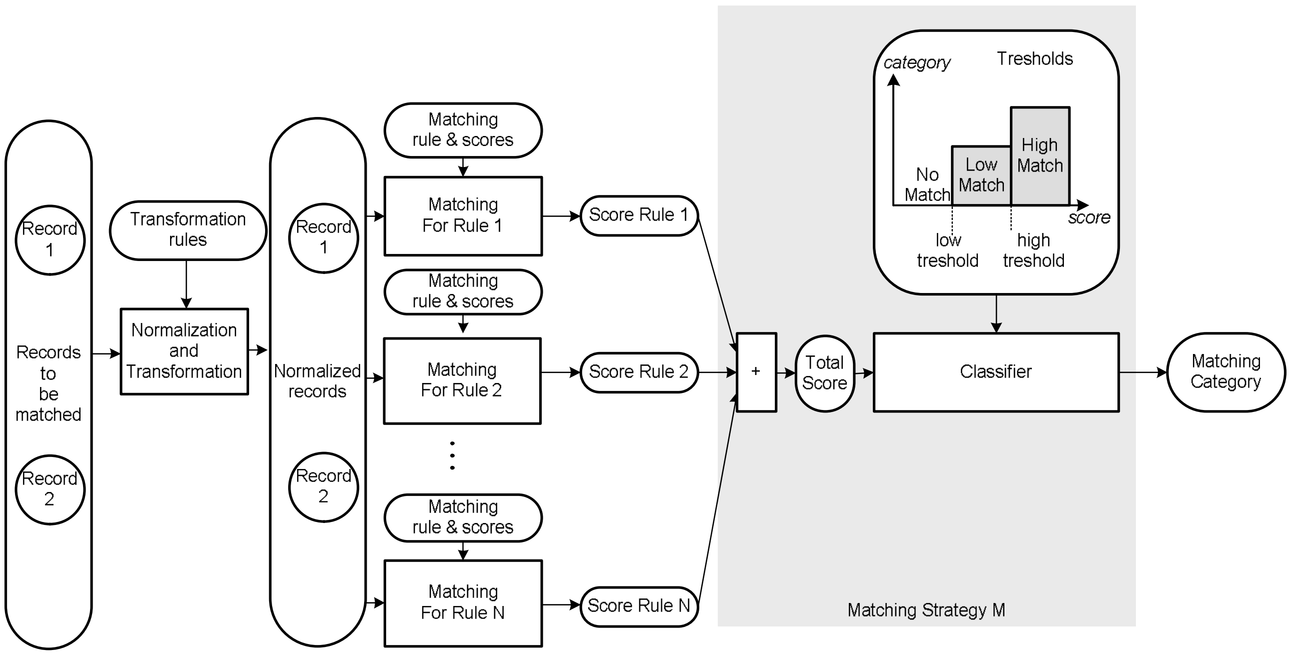


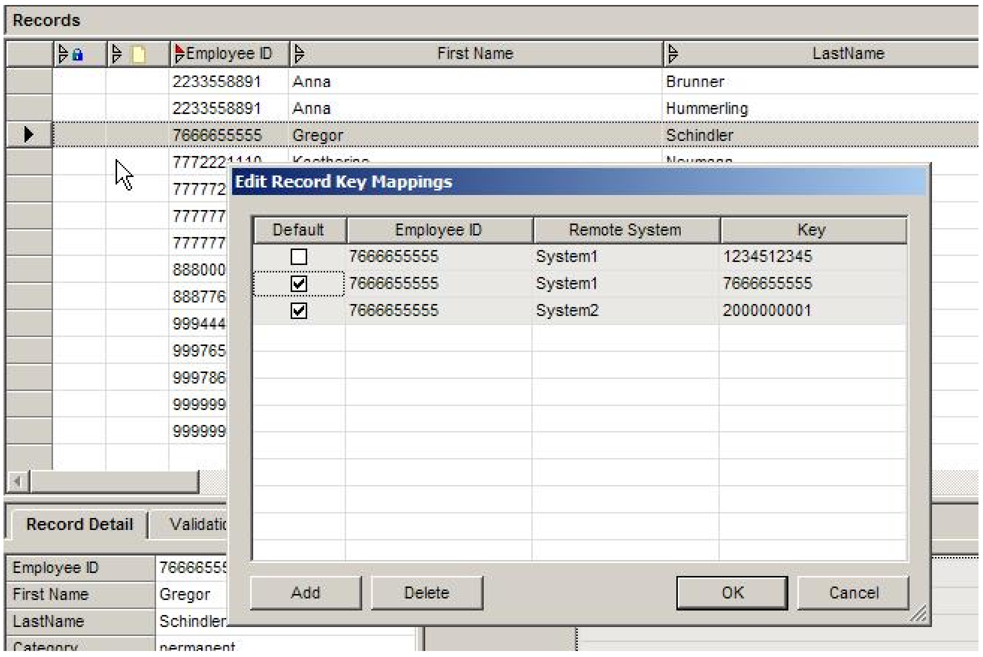
Figure 26 Data Flow for Matching Records

5.6.1.2 Merging and Key Mapping

The records that were identified during matching can be merged in the MDM repository into a new record. Merging typically involves human decisions. Therefore the merging is often defined as a manual workflow step after an automated matching. When multiple records are merged into one, the original records are deleted from the MDM repository. However, the repository still keeps the information about the remote keys of the corresponding records in the remote systems. With this information the attributes of the merged record in the MDM repository can be distributed back to the original records in the remote systems.

Example

The screenshot in Figure 27 shows the key mapping of an employee record that was created by merging three records. Two of them exist as duplicates in the remote system “System 1” and another one exists as an identical in remote system “System 2”. In this example one of the keys in System 1 was used as the Employee ID of the merged record in the MDM repository.



Employee ID	First Name	LastName
2233558891	Anna	Brunner
2233558891	Anna	Hummerling
7666655555	Gregor	Schindler
7772221110	Kaetharina	Neumann
777772		
777777		
777777		
888000		
888776		
999444		
999765		
999786		
999999		
999999		

Default	Employee ID	Remote System	Key
☐	7666655555	System1	1234512345
✓	7666655555	System1	7666655555
✓	7666655555	System2	2000000001

Employee ID	76666555
First Name	Gregor
LastName	Schindler
Category	permanent

Figure 27 Key Mappings For A Merged Record

5.6.2 Expressions, Assignments and Validations

The MDM server includes an expression engine which evaluates expressions that are written in a special MDM expression language. Expressions are used for different purposes:

- Compute the value of **calculated fields**. Calculated fields are read only fields with a value that is calculated from values of other fields of the same or related records.
- Defining **assignments.** In SAP NetWeaver MDM assignments are predefined expressions for mass data maintenance. This is a useful feature for central master data maintenance on the MDM repository or for data cleansing after an import step. With assignments the values of the fields of selected master data records can be set using expressions. This allows for example to set default values for empty fields or to implement automatic correcting to some extent.
- Defining **validations**. Validations are named Boolean expressions that lead to an error if the expression is false. Validations can be invoked manually, automatically on data entry or automatically as a step in an MDM workflow. Validations can be assigned to validation groups that are organized in a hierarchy. With groups a whole set of validations can be invoked together.

An MDM expression is always defined for one “current” record. In the expression the following pieces of data can be accessed: the values of the current record, the values of lookup records (in a nested way), the records associated to the current record via relationships and the original version of a checked out record. In addition it is possible to navigate the hierarchy from a lookup value in a hierarchy table.

The expression language supports arithmetic and logical operators as well as built in functions like string functions and aggregate functions (average, sum, count..). The expression language also supports control flow via if/else. In addition, named expressions can be reused in other expressions.

6 Summary

Master data is the core reference data of an enterprise which must be available for different business processes in good quality. Master data has good quality if it is up to date and free of errors, inconsistencies and undetected duplicates. Master data management (MDM) comprises the definition of processes and responsibilities for master data maintenance, activities and mechanism for master data cleansing and – in case of distributed systems – the synchronization of master data across the enterprise.

MDM is not only a technology but requires also joint efforts of different business areas as well as organizational rules and processes. This broad nature of task is also reflected in software solutions for master data management: On top of a technical MDM infrastructure business content must be provided and application systems must be enabled to participate in MDM tasks and processes. The task of the technical infrastructure is to provide mechanisms for master data distribution, mapping of data structures and values, data cleansing and validation and support for master data related workflows.

There are different approaches for doing MDM in distributed application landscapes. Architectures range from decentralized peer-to-peer models with local responsibility for master data maintenance to architectures with a central MDM hub where master data are collected, cleansed and sometimes even maintained. MDM architectures can also be classified based on the degree of coupling between the application systems (synchronous or asynchronous replication of master data, extent of data that is synchronized).

The SAP NetWeaver MDM product is designed as a generic technology that can be used by customers to implement their specific MDM solution. It contains business content for various SAP master data types from different SAP application systems and an MDM technology consisting of an MDM server, MDM repositories, an MDM Portal UI and a set of MDM clients. Application systems are usually connected to the MDM server using SAP NetWeaver XI. MDM Web services and a rich set of APIs for ABAP and Java can be used to develop custom solutions on the top of the provided functionality.

SAP NetWeaver MDM provides a rich set of functions for mass data maintenance, high speed searching, data matching and data validation. There are some limitations in the area of master data modeling due to the isolation of MDM repositories and the lack of deep hierarchies. As standalone components outside the SAP NetWeaver Application Server the MDM servers, clients and repositories have their own processes for system operation and life cycle management (installation, update etc). As a standalone system SAP NetWeaver MDM can be installed quickly and used in a very flexible way.

The SAP NetWeaver MDM platform is best suited for scenarios with a loose, asynchronous coupling of systems that require the synchronization of only the globally relevant master data and which really make use of the data cleansing capabilities of the product.

7 Further Reading

[0] *Master Data Management: One step at a time*, H. D. Morris, C. W. Olofson, IDC/IBM, White Paper, February 2005, http://whitepapers.zdnet.com/abstract.aspx?docid=178813&promo=999222&kw=Master%20Data%20Managem

[1] *SAP NetWeaver Master Data Management,* L. Helig, S. Karch, Galileo Press, Bonn, Boston, 2007

[2] *SAP NetWeaver Master Data Management 5.5 Documentation*, http://help.sap.com/saphelp_mdmgds55/helpdata/EN

[3] *Magic Quadrant for Customer Data Integration Hubs*, 2Q06; J. Radcliffe ;Gartner RAS Core Research Note G00137774, May 2006

[4] *MDM Master Data Management Hub Architecture, Roger Wolter, Microsoft, MSDN Blog,* http://blogs.msdn.com/rogerwolterblog/archive/2007/01/02/mdm-master-data-management-hub-architecture.aspx

[5] MDM on SAP Developer Network,
general: https://www.sdn.sap.com/irj/sdn/mdm,
discussion forum: https://www.sdn.sap.com/irj/sdn/forum?forumID=55&start=0
weblogs: https://www.sdn.sap.com/irj/sdn/weblogs?blog=/weblogs/topic/14

[6] *SAP NetWeaver Master Data Management Guides* (Installation, Operation, Scenarios), SAP Service Marketplace, http://service.sap.com/installmdm

[7] MDM 5.5 Learning Map, Ramp up Knowledge Transfer, SAP AG http://service.sap.com/~form/sapnet?_FRAME=CONTAINER&_OBJECT=011000358700003471552005E

[8] *Presentations and recoded sessions on MDM in SAP Corporate Portal*
see PTU SAP NetWeaver->Product Information->Key Capabilities->Master Data Management

[9] Master Data Management in AP/A1S, Thomas Vogt, SAP, SAP internal draft paper, May 2007

[10] SAP SRM documentation, MDM-SRM Catalog, http://help.sap.com/saphelp_srmmdm10/helpdata/en

8 Glossary

Term	Definition
Duplicates	The unintended situation that the same real world object is represented by multiple redundant technical objects inside the same system
Identicals	Identicals are objects in different systems that represent the same real life object. Identicals are not a problem but required to represent the same thing in different systems. The goals of master data management is to manage the identicals and to keep their attributes consistent across systems.
Master data	Core data of an enterprise which changes infrequently and is referenced by many business transactions. A precise definition is hard to find, see 3.1.1 for details.
Master data cleansing	Correcting of errors and inconsistencies in master data including the detection and managing of duplicates and identicals
Master data management	Activities, rules and mechanisms that ensure the availability and quality of master data
Syndication	Distribution of master data from the central MDM repository into the application systems.
System of record	The system which provides the single and consolidated view of the master data
System of origin	The system where a piece of master data is created and maintained

9 Appendix

9.1 List of Master Data Objects in AP (Provisional)

The following list is included here only as an example to give the reader an idea about which kind of objects are considered as master data in SAP applications. This is a provisional list is taken from an SAP internal draft paper [9]. It does not claim to be complete, is not approved as an AP guideline and must not be used as a reference for AP master data.

.

No	Business Object	Description
1	Access Control List	An Access Control List is a list of access groups that have access to the entire host object during a validity period.
2	Access Group	A group of identities for which access control is specified in a certain context.
3	Bank Directory Entry	An entry for a bank in a directory of banks.
4	Bill of Exchange Storage	A company's storage for bill of exchange receivables in a currency.
5	Business Partner	A person, an organization, or a group of persons or organizations, in which a company has a business interest.
6	Business Plan Variant Specification	A specification of external business conditions, business assumptions, business goals, and procedures of data collection for a business plan. For example, different business variant specifications based on optimistic, conservative, or pessimistic assumptions can be used in parallel.
7	Cash Storage	A company's storage for cash of a currency. A Cash Storage contains: o The description of the storage, an allowed currency o Information about the physical location (address) of the storage
8	Cheque Storage	A location for incoming checks that a company receives from its business partners, such as customers.
9	Clearing House	An organization that provides services for the settlement of payment card payments, such as the authorization of payments.
10	Clearing House Account	A company-internal representation of an account that is set up and managed at a clearing house for card payments for the company. It is based on an agreement between the company and the clearing house.
11	Communication Arrangement	An arrangement between the system owner and a Business Partner, containing the communication settings for XML messaging, file-based communication and form-based communication via output channels like print, email and fax.
12	Company	A financially and legally-independent organizational center that is not tied to a geographical location and that is registered under business law.
13	Company Tax Arrangement	An agreement between a company and a tax authority regarding the declaration and payment of taxes.
14	Compensation Component Type	A description of the employee compensation components in the context of Human Resources.
15	Compensation Structure	An organized structure of pay grade ranges. A pay grade range reflects the value of tasks and activities in the company. Employees can be assigned to a pay grade range based on the tasks and activities they perform. A Compensation Structure can be company-specific or can be predefined according to pay scale regulations.
16	Core View Of Cash Storage	Core View of a Cash Storage
17	Core View Of House Bank Account	A view of a house bank account that contains identification, type, and currency data.
18	Cost Centre	An organizational center where costs are incurred and for which costs are recorded separately.
19	Customer	A business partner to whom materials or services are offered or provided.

20	Customer Problem and Solution	A collection consisting of one or several problems reported by a customer, and one or several solutions provided by one or more experts.
21	DE_Employee Social Insurance Arrangement	An arrangement for the employee by the German bodies that are legally responsible for administering the employee's social insurance contributions and benefits. This arrangement concerns the information required for calculation of German social insurance contributions and reporting according to the German Data Entry and Transfer Regulation ("DEUEV", Datenerfassungs- und Übermittlungsverordnung).
22	DE_Employee Tax Arrangement	An arrangement by the German tax authority for the employee, concerning calculation and reporting of income tax deductions according to German legal requirements.
23	Document	A carrier of unstructured information and additional control and monitoring information.
24	Employee	A person who contributes or has contributed to the creation of goods or services for a company. Employee includes both internal and external employees (service performers). Unlike externals, internal employees are bound by instructions and obliged to adhere to the company's policies and regulations.
25	Employee Compensation Agreement	An agreement between an employer and an employee detailing all compensation components that are relevant to the employee, such as base salary, one-time and recurring payments and payments for employee benefits. Also part of this agreement can be the assignment of a Compensation Structure Grade which shall be valid for the employee.
26	Employee Payroll Agreement	An agreement between the employer and employee concerning the personal conditions for payroll processing. This agreement determines the fundamental differentials of the payroll behavior for the employee.
27	Employee Time Agreement	An agreement between employer and employee consisting of time management stipulations that are derived from legal, company-specific, and pay-related provisions, and from terms agreed individually with the employee.
28	Employment	A relationship that comes into being by virtue of one or more valid work agreements. Whereas the work agreement consists only of the specific labor-related arrangements agreed between company and employee, the employment encompasses the entire legal relationship between the contracting parties.
29	Equipment Resource	An Equipment Resource is a permanently installed operating facility or a group of identical operating facilities that has the capacity to provide services.
30	Exchange Rate	The relationship in which one currency can be exchanged for another currency at a specified time.
31	Expense Arrangement	A definition by the company of parameters for an employee that are needed for expense reports. It is used to determine per diem amounts, the type of standard vehicle, and the standard cost distribution.
32	Functional Unit	An organizational center that is responsible for the planning, execution, and administration of business process steps. The responsibility can lie with the organizational center itself or it can be delegated to other organizational centers.
33	House Bank	An organization that provides banking services for the company, such as account management or lockbox.
34	House Bank Account	An internal representation of a company's bank account at a house bank. It contains all control information required for processing payment-relevant processes.
35	Identified Logistic Unit	A physical unit existing exactly once in the real world, which is identifiable for logistic purposes. A Identified Logistic Unit describes the logistics and physical aspects of a product or package.
36	Identified Stock	A subset of a material that shares a set of common characteristics, is logistically handled separately from other subsets of the same material and is uniquely identified.

37	Identity	An identity is a representation of the uniqueness of a human person or non-human subject in a uniform way. The identity specifies the person's or subject's credentials for accessing systems in a system landscape, the granted authorizations and the system settings which are valid for the person or subject.
38	Individual Material	A tangible product that occurs only once in the real world and is therefore uniquely identifiable.
39	Individual Material Service Process Control	A process-driven view that contains information about an individual material that is required to use the individual material in customer service processes.
40	Inspection Rule	A rule that contains the specifications for the inspection. The inspection is used to check whether an object or procedure meets predefined requirements.
41	Installation Point	A physical or logical location at which a business object, for example software or a material, is installed during a certain period of time. An installation point contains descriptive information about its installed object, for example, the quantity of materials used, and can be structured in a hierarchical relationship with other installation points.
42	Installed Base	An Installed Base is a container that holds structured information of business components and their compositions as well as their business features. Installed Base Components carry properties of business objects (e.g. Material or Individual Material), which have been assigned to an Installed Base. They can be multi-level structured, are time dependent and contain descriptive information about their corresponding business component. Content of an Installed Base Component might for instance be: Address and/or application specific extensions.
43	Job	The type of a position.
44	Labour Resource	An employee or a group of employees with the same skills and qualifications with the capacity to operate specific devices or to perform specific tasks.
45	Location	A geographical place.
46	Logistic Unit	An item established for logistics operations, such as storage, movement, and packing. A Logistic Unit represents all physical units handled in the same manner during logistic operations, whether they are packed or unpacked goods.
47	Logistic Unit Usage	A logistics purpose for which Logistic Units are grouped. The Logistic Unit Usage can represent a process or an activity, such as conveying, packing, or storing.
48	Logistics Area	A freely definable area within a location providing detailed physical and operational information required for storage and production. Logistics areas can be arranged in a hierarchy according to physical aspects or logistical functions.
49	Logistics Break Program	A set of breaks in supply chain processes such as production, warehousing and transportation that are either scheduled at an absolute time of the day or scheduled relative to the start time of a shift.
50	Logistics Shift	A period of working time (called shift) in supply chain processes such as production, warehousing, and transportation that can be interrupted by breaks.
51	Logistics Shift Program	A set of shifts, organized as a generic program, in supply chain processes such as production, warehousing and transportation that span over a period of time.
52	Logistics Source and Destination Determination Rule	A rule for determining the logistics source for inventory retrieval or the logistics destination for inventory placement for a specific logistics area or location.
53	Logistics Task Folder	A folder for storing and grouping logistics tasks according to business criteria. Logistics Task Folder contains details about the processors that are registered at the folder.
54	Material	A tangible product, such as a sellable article, packaging, auxiliary material, and expendable supplies, that can be created and then represents a business value. A material can be traded, consumed, or produced.

55	Material Availability Confirmation Process Control	A process-driven view that contains information about a material that is required to use the material in availability confirmation processes.
56	Material Availability Confirmation Process Control Template	A template that simplifies the maintenance of material master data needed to execute the availability check.
57	Material Financials Process Control	A process-driven view that contains information about a material that is required to use the material in financial processes.
58	Material Financials Process Control Template	A template that simplifies the maintenance of material master data needed to processes in Financials (including the inventory valuation).
59	Material Inventory Process Control	A process-driven view that contains information about a material that is required to use the material in logistics processes.
60	Material Inventory Process Control Template	A template that simplifies the maintenance of material master data needed to control several logistic processes. The collection includes the inventory management unit of measure and the unit of measure in which a material is serialized.
61	Material Procurement Process Control	A process-driven view that contains information about a material that is required to use the material in procurement-relevant processes.
62	Material Procurement Process Control Template	A template that simplifies the maintenance of material master data needed to use the material in procurement-relevant processes.
63	Material Sales Process Control	A process-driven view that contains information about a material that is required to use the material in presales, sales, and customer service processes.
64	Material Sales Process Control Template	A template that simplifies the maintenance of material master data needed to control presales, sales, and customer service processes.
65	Material Supply Planning Process Control	A process-driven view that contains information about a material that is required to use the material in supply planning processes.
66	Material Supply Planning Process Control Template	A template that simplifies the maintenance of material master data needed to control procurement planning. Planning is performed at supply planning area level (including specifications for lot-size planning, requirement consumption, in-house production and external procurement).
67	Material Template	A template that contains material data relevant for presales, sales, customer service, financials, supply planning and availability confirmation processes.
68	Material Valuation Data Template	A template that contains material valuation data relevant for processes in Financial Accounting.
69	Organizational Center	A business unit in an organizational structure of a company within the extended enterprise.
70	Organizational Center Template	A collection of pre-defined information used to create a new Organizational Centre. It is used to facilitate the creation of new Organizational Centers which have several attributes in common.
71	Packing Bill of Material	A complete and structured list of components that defines the packing structure of logistic units.
72	Payment Agreement	An agreement between a company and a business partner on the handling of payments. It defines, for example, the payment methods allowed and which bank details or credit cards should be used.
73	Payment Card	A card that facilitates the card owner to purchase goods or services from merchants or companies affiliated to clearing houses, without cash payment.
74	Permanent Establishment	An organizational center that represents a corporate division that is tied to a geographical location and whose business activity is subject to uniform legal and fiscal processing.
75	Position	An organizational element within the organizational plan of an enterprise. It comprises a fixed combination of tasks, competencies, and responsibilities that can be taken care of by one or more appropriate employees.

76	Procurement Arrangement	An arrangement between a strategic purchasing unit and a supplier that is used for procurement transactions. The arrangement can also be established for one supplier across all purchasing units. This arrangement contains, for example, payment terms, invoice currency, and incoterms. This arrangement does not constitute a contract with the supplier.
77	Procurement Price Specification	The specification of a price, a discount, or a surcharge for procurement of goods or services. The specification is defined for a combination of property values and is valid for a specific period.
78	Product Catalogue	A structured directory of catalog items, where each catalog item represents a product and provides information about it.
79	Product Catalogue Update Rule	A set of rules that controls how products are cataloged in a product catalog and how to update the product catalog when cataloged products are changed.
80	Product Category Hierarchy	A hierarchical arrangement of product categories according to objective business aspects. Subordinate product categories represent a semantic refinement of the respective higher-level product category.
81	Production Bill of Material	A complete and structured list that defines and describes the components that are required in the production of a material including all its variants.
82	Production Bill of Operations	A description of a production process for manufacturing a product. It contains all the processing or transformation steps that have to be executed. It also defines all the resources to be used with all the necessary technical specifications such as the standard times, capacity requirements, and work instructions.
83	Production Model	A model of a production process that is defined by a network of production segments.
84	Production Segment	A part of a production process specified by a network of operations and assigned materials for the production of a material.
85	Profit Centre	An organizational center that represents an area of a company for which a separate period-based profit is determined and used for profit-oriented analysis and management of the area's activities.
86	Project Template	A project template defines the structure and non-operational data of a project. It is used for a standardized project planning and execution – a new project may be generated from a project template.
87	Published Product Catalogue	A version of a product catalog that has been released for access by, or exchange with, the target group for which the contents of the product catalog have been tailored.
88	Quality Issue Category Catalogue	A structured catalog of categories that classifies quality-related issues for a particular quality aspect.
89	Released Execution Production Model	A released version of a production model that contains all the production bill of operations and production bill of material data required for the execution of a production process.
90	Released Planning Production Model	A released version of a production model that contains all the production bill of operations and production bill of material data required for the planning of a production process.
91	Released Site Logistics Process Model	A released version of a site logistics process model that contains all elements required for defining and describing the execution of a site logistics process.
92	Reporting Line Unit	An organizational center in the personnel reporting line of a company. A reporting line unit typically has a personnel manager who is responsible for defining the objectives and salaries of the directly or indirectly-assigned positions and employees.
93	Resource Group	A grouping of individual resources that provide similar services or have similar physical and functional characteristics.
94	Resource Operating Time Template	A template of an operating time definition that contains all information required to maintain the operating times for multiple resources.
95	Responsibility	A responsibility describes specific rights and duties of an acting agent responsible such as a person or an organizational centre etc.

96	Sales Arrangement	An arrangement between a sales organization and a customer that is used for sales transactions. This arrangement contains, for example, payment terms, invoice currency, and incoterms. This arrangement does not constitute a contract with the customer.
97	Sales Price List	Sales Price List is a combination of specifications for prices, discounts or surcharges, (Price Specification), in Sales and Service. The list is defined for a combination of properties, and is valid for a specific time period.
98	Sales Price Specification	Sales Price Specification is the specification of a price, a discount, or a surcharge for sales and service. The specification is defined for a combination of properties and is valid for a specific period.
99	Sample Drawing Procedure	A procedure that defines how samples are to be taken. It contains data for the sample-drawing frequency, sample size, and sample quantity.
100	Segment	An organizational center that represents a corporate division for which financial information has to be published. Its business activities result in revenue and expenditure, and the operating income is regularly monitored by the main decision-makers for the purpose of assigning resources and evaluating performance.
101	Service Issue Category Catalogue	A structured directory of issue categories that group business transactions in Customer Service from an objective or a subjective point of view.
102	Service Level Objective	A Service Level Objective (SLO) is a measurable objective that specifies one or more conditions for fulfilling a service.
103	Service Product	An intangible product that describes the provision of a service. A service is provided at the time of its use.
104	Service Product Procurement Process Control	A process-driven view that contains information about a service that is required to use the service in procurement-relevant processes.
105	Service Product Financials Process Control	A process-driven view that contains information about a service that is required to use the service in financial processes.
106	Service Product Financials Process Control Template	A template that simplifies that simplifies the maintenance of Service Product master data needed to processes in Financials (including the inventory valuation).
107	Service Product Procurement Process Control Template	A template that simplifies that simplifies the maintenance of Service Product master data needed to use the Service Product in procurement-relevant processes.
108	Service Product Sales Process Control	A process-driven view that contains information about a service that is required to use the service in presales, sales, and customer service processes.
109	Service Product Sales Process Control Template	A template that simplifies the maintenance of service product master data needed to control presales, sales, and customer service processes.
110	Service Product Template	A template that contains service product data relevant for presales, sales, customer service and financials processes.
111	Service Product Valuation Data Template	A template that contains service product valuation data relevant for processes in Financial Accounting.
112	Set of Books	A collection of specifications structuring a body of accounting records containing all data of all positions of a balance sheet and profit and loss statement. It comprises specifications for the chart of accounts, the fiscal year variant, the accounting principle, and the company currencies.
113	Site Logistics Bill of Operations	A description of a process for the company-internal movement of goods, goods receipt, or goods issue. It defines any inventory changes with regard to location, structure, or quantity that may arise as a result of unloading, unpacking, or transportation. It also defines all the resources, Logistic Units, standard values, and all work instructions to be used.
114	Site Logistics Process Model	A model of site logistics process that is specified by a sequence of site logistics process segments.
115	Site Logistics Process Segment	A part of a logistics process specified by a net of operations for packing, moving and checking of goods.
116	Source of Supply	A source for the internal and external procurement and the internal production of one or more products.

117	Storage Behaviour Method	A set of rules that defines the manner in which a storage location is managed.
118	Supplier	A business partner who offers or provides materials or services.
119	Supplier Product Category Life Cycle	A representation of the supplier life cycle that contains information about the status of supplier development and the product category delivered by that supplier.
120	Supplier Transaction Assessment	The assessment of a supplier's performance based on business transactions that are evaluated automatically.
121	Supply Planning Area	An area for which a separate planning ensures the availability of products on time.
122	Supply Quota Arrangement	An arrangement that specifies how material demands or material issues are distributed to different sources of supply, business partners, or internal organizational units.
123	Tax Authority	An authority that collects and administers taxes.
124	Trade Receivables Payables Account	An account of all trade receivables and payables of a company from or to a business partner. It also contains guidelines and agreements concerning the payments and dunning of receivables and payables for a business partner.
125	Transportation Lane	A relationship between two locations or transportation zones that specifies the materials that can be transported between locations or transportation zones, and the means of transport that can be used.
126	Transportation Zone	A zone containing geographical locations that may be considered collectively for modeling or planning transportation routes or transportations.
127	Vehicle Resource	A means of transportation or a group of identical means of transportation that has the capacity to provide transportation services.
128	Warranty	A guarantee to vouch for defects or faults in the product purchased that is valid for a specific period of time. The type and scope of these services, such as repairing a defect for free or taking the product back, are defined in the warranty.
129	Warranty Service Process Control	A process-driven view that contains information about a warranty that is required to use the warranty in customer service processes.
130	Work Agreement	A contract between employer and employee that obligates the employee to provide his or her labor and the employer to provide the agreed compensation.
131	Working Time Model	An employee-independent, structured description of working times. In addition to working times, it may also describe absence times, break times, and availability times.
132	Working Time Model Catalogue	Structured directory of the available working time models. The availability can be restricted to parts of the organization for individual parts of the catalog.