MarkLogic®

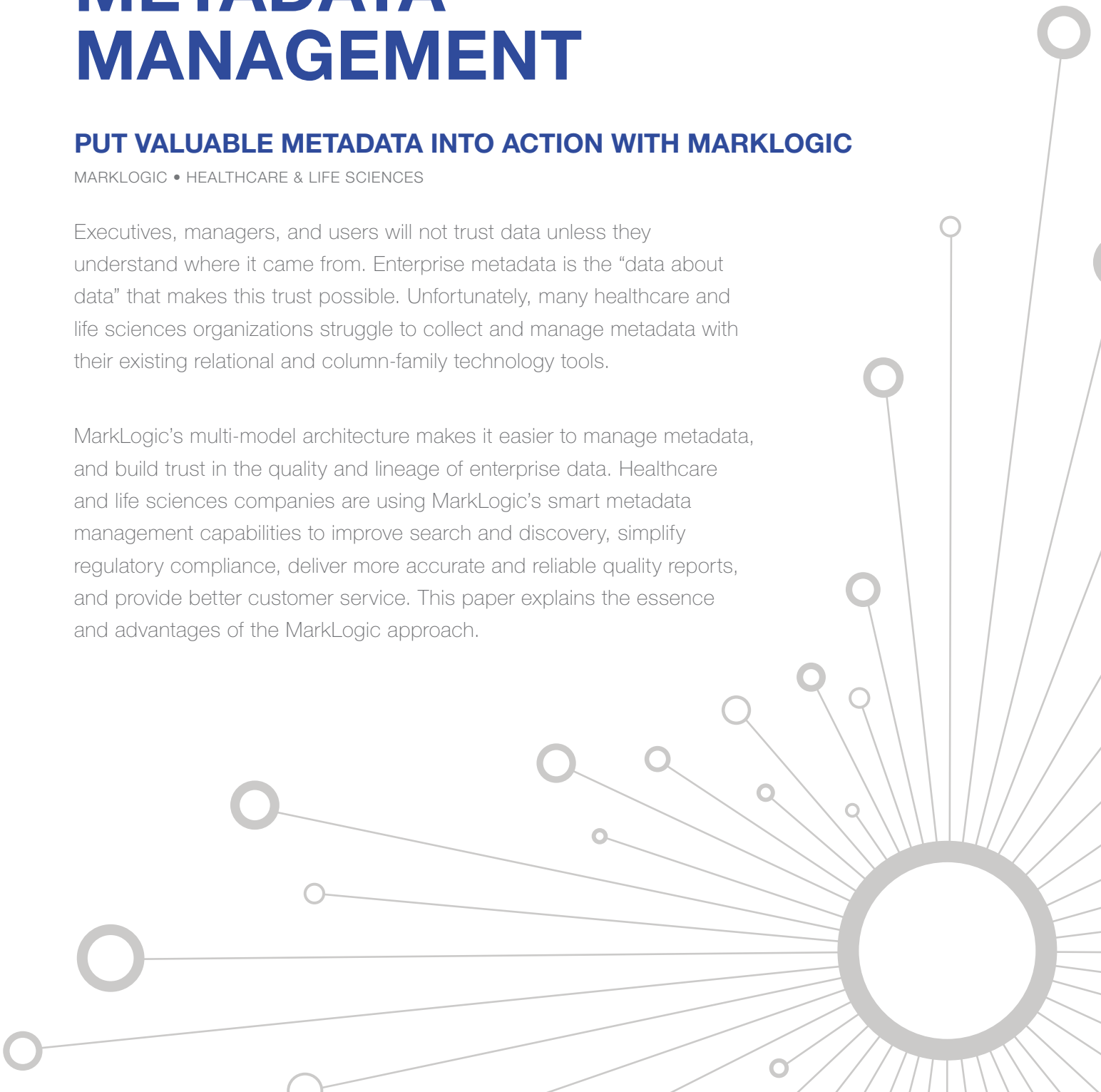
# AGILE ENTERPRISE METADATA MANAGEMENT

## PUT VALUABLE METADATA INTO ACTION WITH MARKLOGIC

MARKLOGIC • HEALTHCARE & LIFE SCIENCES

Executives, managers, and users will not trust data unless they understand where it came from. Enterprise metadata is the "data about data" that makes this trust possible. Unfortunately, many healthcare and life sciences organizations struggle to collect and manage metadata with their existing relational and column-family technology tools.

MarkLogic's multi-model architecture makes it easier to manage metadata, and build trust in the quality and lineage of enterprise data. Healthcare and life sciences companies are using MarkLogic's smart metadata management capabilities to improve search and discovery, simplify regulatory compliance, deliver more accurate and reliable quality reports, and provide better customer service. This paper explains the essence and advantages of the MarkLogic approach.



## ABOUT THIS DOCUMENT

This document describes the advantages of using MarkLogic to manage enterprise metadata. It:

- Defines enterprise metadata and examines its ubiquity, variety, and importance
- Presents the diverse requirements for enterprise metadata management
- Explains why MarkLogic's multi-model (document and graph) architecture is ideal for coping with the high variability of enterprise metadata
- Compares MarkLogic's metadata management capabilities to those of less-flexible data architectures
- Shows how MarkLogic's universal indexes, search, and semantic features further simplify metadata management

The document concludes by looking at some MarkLogic use cases for metadata management, and draws some conclusions about how MarkLogic can dramatically increase enterprise agility.

**Intended Audience:** This document is intended for anyone trying to understand the importance of enterprise metadata and architectural approaches to managing it. It assumes minimal familiarity with various database concepts such as relational, key-value, graph and document stores.

## WHAT IS ENTERPRISE METADATA?

One of the challenges of effective metadata management is clearly defining metadata. The general definition "data that describes other data" is a starting point, but is too broad to be very helpful.

We suggest polling different parts of your organization and asking them what metadata means to them. When we do this exercise with our clients, we find that different business units and job functions have radically different understandings and requirements. Our survey typically reveals different combinations of these metadata management needs:

- A **business glossary** of all the acronyms and terms that they use in each project
- A **taxonomy** or ontology used to classify items in collections
- The **data models** for their various databases including tables, columns, data elements, document structures, and definitions
- The **reference data** or codes that they use to control data classifications and convert numeric codes into human-readable labels
- Their **business rules** for validating data, transforming data, classifying data linking data and assigning **data quality scores** to each record
- Their **workflows** and approval processes
- A list of their **reports**, dashboards, and key performance indicators, and calls to Application Program Interfaces (APIs)
- **"Operational"** metadata which includes how long various reports typically take to run and how much CPU, memory and disk resources they consume, and any chargeback and billing associated with these functions
- **Relationships** between records, including different representations of the same customer in different systems (the discipline of maintaining these relationships is often called Master Data Management or MDM)
- Complete knowledge of where these artifacts came from (**who, why, when**), the **relationships** between them, and how they are maintained and **tested**

When we get these surveys back, we are usually surprised by the diversity of the requirements across departments and domains. However, we often quickly see patterns emerging. Here is a sample of common findings:

- Some organizations' metadata needs to be carefully **controlled** by their designated data stewards. This is important as small changes in metadata structures can have large impacts on production systems.
- Some organizations' metadata needs to be **versioned or have effective dates**. Clients need to support multiple concurrent production versions of metadata and be able to undo changes to production systems without interruptions in service.
- Some organizations' metadata systems such as reference-code lookups and user interfaces with selection lists need to have **high availability** and

**real-time** lookup and search.

- Some organizations' metadata users need to be able to quickly find and remove **duplicate** records so that we have a single representation of common meaning.
- Some organizations' metadata is structured and some of it is unstructured. For example, this may include traditional spreadsheets of data, as well as complex graphs and other relationship-oriented data.
- In many cases, organizations need to have a clear focus on the meaning of their data and the **semantics** of individual data elements and codes.
- Each business unit may need to **customize** the way it handles metadata, yet the organization needs **consistency** across the enterprise to look at the data inter-departmentally or enterprise wide.

### MARKLOGIC'S APPROACH TO METADATA MANAGEMENT UTILIZES KEY ENTERPRISE FEATURES:

- Flexible Metadata Storage
- Search Enablement
- Real-time Data Services
- Enrichment and Annotations
- Semantics (RDF Triples)
- Data Provenance and Governance
- Fine-grained Security
- Reporting/Analytics
- Scalability

If you have previously come to the conclusion that **no single data technology or application easily manages all these requirements**, you are not alone! At MarkLogic we have found that the diversity of data types and diversity of **uses** of metadata means that we need the most flexible, agile, secure and reliable software environment possible.

At this point, you might completely disagree with your users on what metadata is or is not. That can happen and can be a frustrating experience. But stay mindful of the goal: to manage this type of information in a secure and consistent way across the enterprise, so that metadata assets can be found and reused, thus maximizing the business value of their data assets. If an organization uses spreadsheets to manage business-critical information, you might ask if there is a better way of storing and using these datasets. This is a very good question to be asking!

Next, let's clearly define what this diversity of metadata is and how MarkLogic helps to manage this type of information.

## THE CHALLENGES OF HIGHLY VARIABLE METADATA

Let's take a look at a simple problem of who created and updated a record, and see how data variability comes into play. We will see how traditional relational systems lack the flexibility that we need to manage this information.

Many of us have seen four "history" columns added to the end of a row of data. For example:

| CREATED_BY | CREATED_DT | UPDATED_BY | UPDATED_DT |
|---|---|---|---|
| sperterson | 2016-02-25 | fbrown | 2016-03-12 |
| rbatchu | 2015-01-15 | rbatchu | 2016-03-12 |

Pretty simple, right? Each row in a table just needs four columns. Just add those columns to every table in our relational database, add some code to update the columns, and we are off and running!

Oops, but wait! What if we also need to track not just the first and last change, but also all the intermediate changes? In a relational model we can create a separate history table and do a JOIN operation. This will slow down our queries, but it is a bit more flexible.

Suppose we also want to keep a copy of the older versions of the record. What if we want a searchable full-text description of why each change was made? What if we want to classify the record with keywords and assign a data quality score to the row? What if we want to see who approved each change and when the approvals were done? We cannot do this by simply adding a few columns to every table. What if only a few rows out of millions need this additional data? Do we leave placeholders for each row? What if we need to treat business objects that are shredded across multiple rows as a single traceable entity?

> “What tables and relational systems lack is the ability to attach high-variability metadata without disrupting records that do not need this information.

Now we start to appreciate some of the challenges with trying to store complex and highly variable metadata in tabular or relational structures. If you have a single item that needs additional metadata, then every row in that table *must* have that item added. What tables and relational systems lack is the ability to attach high-variability metadata without disrupting records that do not need this information. In short, we need a more flexible approach.

Next, let’s take a look at the options beyond simple tabular data stores.

## ARCHITECTURE OPTIONS FOR METADATA MANAGEMENT

Below are the six main data architectures that we have available to us:

1. Relational
2. Analytical (OLAP Cubes)
3. Key-Value Stores
4. Column Family Stores
5. Graph Stores
6. Document Stores

These six database architectures have been around for a long time (even on mainframes) and we are not really seeing new variations that are changing the way we manage data.

We can quickly discard analytical systems for metadata management. Analytical systems are intended primarily for doing historical reporting over aggregates and their star-schema pattern depends on a central fact table that has little variability.

We can also remove both key-value stores and column family stores, since both lack query languages that are flexible enough to index, query, and search arbitrary metadata. Many of these key-value systems are really designed for scalability, not semantics or agility.

That leaves us with three options: relational, graph and document stores. But we already showed how relational systems and tabular models (fixed column and row) do not handle high-variability well. So let’s now do a deep dive into how using MarkLogic’s multi-model (document and graph) architecture is ideal for storing and searching enterprise metadata.

### HOW DOCUMENT (AND GRAPH) STORES MANAGE VARIABLE METADATA

Many people are familiar with JSON or XML files. They can be thought of as tree-structures where the data payload is carried in the leaves of each branch. Documents and graphs all have a “recursive” property in that all branches in a document can contain sub-branches, which in turn can also contain sub-branches. Graphs have similar properties. A graph can always point to another graph that contains additional metadata. But for now, let’s focus on documents and see how our row-history is represented.

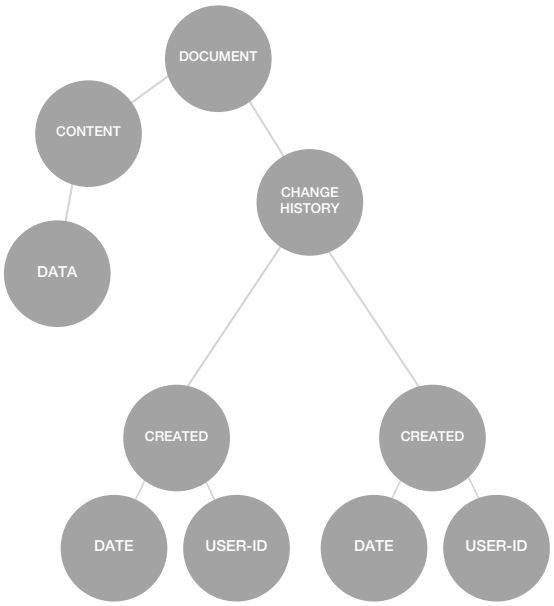



Figure 2. A change history branch is added to a document

This is a key difference between a relational architecture and document architecture. In the relational model, if you add a single column to a table, you have to add this column to every row in the entire table. If you have 1 million rows, a single new column adds 1 million new cells to your table. There is no flexibility here.

Let's contrast this to a document store. With a document store any branch in any document can have new branches added to it any time. We have complete flexibility about how documents are extended, even if it is for one out of a million documents. In summary, document (and graph) stores are the ideal ways to store highly variable metadata. They win because they are more flexible.

Once you start to store your metadata in a document store, you might find that using tables to store metadata is like wearing a straightjacket. Using a document store is a fundamental factor that keeps your metadata management agile and allows you to quickly customize metadata solutions to meet the diverse needs of every business unit.

You might now be wondering: if every business unit adds their own metadata to every item in my database, won't we have chaos? Won't our reports all stop working? The answer to this is sometimes.

## COUNT THE WAYS TO STORE METADATA IN MARKLOGIC

Because MarkLogic stores data as both documents (JSON and XML) and graphs (RDF), you might guess that there are flexible ways to store your metadata. Below are a few of the most popular examples:

1. Use a "wrapper" or "envelope" element to bundle both document-data and metadata in a single document
2. Use the MarkLogic "properties" APIs to store metadata such as last-modified times
3. Use in-document RDF triples to store metadata
4. Use external RDF to link a document's URI to its metadata
5. Use a separate document or documents to store document metadata. This is useful when metadata security roles are different from the source document
6. Use key-value pair metadata (maps) to store metadata (MarkLogic 9)

Without strong data governance and controls in place, document and graph stores can become difficult to manage. However, it turns out that MarkLogic has some powerful tools to help you manage data governance and security.

> Using a document store keeps your metadata management agile and allows you to quickly customize metadata solutions to meet the diverse needs of every business unit.

This leads us into discussing some techniques that you can use MarkLogic for to detect and correct variations in both your primary data as well as your metadata. Our first line of defense is something called a Universal Index. Let's see how this works.

## HOW MARKLOGIC'S UNIVERSAL INDEX HELPS MANAGE HIGHLY VARIABLE METADATA

To this point, we have talked about how document stores allow us to store highly variable data. What is important to note is that the MarkLogic database platform immediately indexes each branch (content *and* structure) of every document as soon as it is added (ingested or loaded). This index – called the *Universal Index* – can be used to quickly report on all the variations in your data.

The history example previously mentioned is a good place to start. Let's assume that most records only need metadata on the user that created and modified the records as well as the dates and times they made these changes. We will also assume that there are some special records where we need to get a manager's approval before we change them. You can support this workflow with MarkLogic by writing a query that will force all changes to only become current if these special records have a manager's approval.

You can also use MarkLogic's powerful schema validation tools to check that these rules have been followed. MarkLogic has a robust set of tools for checking that any document conforms to a set of data quality rules. These rules are defined in a document called an "XML Schema." Each document can be tested against the XML Schema and a numeric data quality score (for example 1 to 100) assigned to each document. You can use data quality scores to filter out bad data or create a list of documents that might need cleanup by automated processes or real people.

MarkLogic also allows you to associate a document quality score to each document within the database. This score is typically an integer between 1 and 100. The validation process can be used to change the quality score, which can then prevent the document from appearing in reports or can change the ranking of a document in a search result.

You can have both high variability and consistency of rules across all enterprise data. No relational models offer this same flexibility and power.

And, MarkLogic's indexing capabilities are not limited to just a single index. In addition to the Universal Index, there are many types of indexes available for you to use to optimize your queries. For example, a range index can be added any time you have ordered information such as dates. Thankfully, you do not need to be an expert on all these indexes to set up your first MarkLogic metadata management system. MarkLogic provides a set of default settings that will usually be sufficient for many projects.

Now we are going to take a closer look at one of the most difficult problems in metadata management: creating trusted systems by removing duplicate metadata entries. Many organizations find that their users do not trust their metadata management technology, so they do not use it. Worse yet, they may create their own departmental solution if they do not trust or cannot customize the enterprise solution.

### METADATA REPOSITORY VS. METADATA REGISTRY

Many organizations work hard to collect all the various metadata elements used in each system of each of the business units in a single location. They use tools that periodically scan each database for items such as the table names, views, column names and relationships. These systems can then help people find the right fields they might need when they are running reports. We call these systems "Metadata Repositories" or "Metadata Catalogs" since they focus on collection and search, but not the task of promoting a consistent use of metadata.

Another distinct task is to create systems that recommend what terms or data elements must be used to conform to enterprise data standards. These are called "Metadata Registries". These systems are curated publishers of enterprise standards. The role of a trained "data steward" is to manage data elements in these systems. Registries focus on search, de-duplication, and carefully orchestrated approval processes guided by an approved data governance process.

## BUILDING TRUSTED ENTERPRISE METADATA MANAGEMENT SYSTEMS

Here is a simple problem to begin. You want to have one, and only one, way to represent a person's birth date in your system. You start off by adding a data element called **PersonBirthDate**. You ask everyone to use this as the column name, XML element name or JSON name. You setup your registry and go on vacation for a few weeks. When you come back, you find that a user was searching for **IndividualDOB** and didn't find your element. So they added their own. Another did not like either one so they added **CustomerBirthDate**. Now when people create a table they use whatever name for birth date they feel like because they were confused about what version to use.

Duplicate data definitions can be one of the most trust-busting aspects of any metadata management system. If users find two definitions on how to store a birth date, which one should they use? Perhaps they should create their own new data element?

This story shows how critical the role of **search** and approval processes is when building a metadata management system. If the search tool knew that "Person" and "Individual" were related terms – and if users were not allowed to arbitrarily create their own registry elements – then we would have avoided duplicating the definitions.

### USING MARKLOGIC'S ROLE-BASED ACCESS CONTROL TO MANAGE METADATA

Metadata registries specifically focus on the concept of access control to administered items. An administered item is any data that will impact other systems if they are not controlled. Administered items need to define what roles can view and update these items.

MarkLogic has a flexible system for associating multiple roles to every document in the metadata registry. This is called Role-Based Access Control (RBAC). When a new user is added to the system, they are assigned to one or more roles. Every query then checks to see what roles are allowed to update documents before updates are done.

> "Using MarkLogic's Role Based Access Control allows organizations to carefully control and audit metadata repositories.

Let's take the example of an organization that has a team of people that define the data standards for all healthcare claims. A metadata registry can create a role called "claims-metadata-manager". All terms and data elements can be associated with this role. Users might be granted this role only after they have gone through a training process to understand the impact of their changes on your systems. Any users that do not have this role might be able to view these documents, but not be able to change the documents.

Using MarkLogic's Role Based Access Control allows organizations to carefully control and audit metadata repositories.

### USING MARKLOGIC VALIDATION AND DATA QUALITY TOOLS TO MANAGE METADATA

Data Stewards might also be responsible for maintaining high quality data standards within their domain. For example, a data steward responsible for claim metadata might be responsible for defining rules that validate various forms of documents and assigning a data quality score (e.g., a number from 1 to 100) to each claim document. Claims that are incorrect, incomplete, or being reviewed might be assigned a lower score. Your reporting system might only include claims with a score of 70 or higher.

One of the challenges with using tabular systems to manage data quality is that they do not have a single uniform concept of a data quality score across all their systems. Data quality is often an ad-hoc process and each business unit may use different strategies to validate and score data. Very often changes in a single reference table can have ripple effects that impact the data quality scores of millions of records.

MarkLogic easily handles this problem by storing all data quality information as a single metadata property in all documents. All of the MarkLogic functions use this property to store data quality. You do not need to build any additional tools to validate documents from XML schemas and to create special tables to store your quality metrics. MarkLogic comes ready to manage data quality out-of-the box.

> "One of the challenges with using tabular systems to manage data quality is that they do not have a single uniform concept of a data quality score across all their systems.

Now that we have an overview of how the core features of MarkLogic are used to manage enterprise metadata, let's look at some detailed examples of how healthcare and life sciences organizations put these features together to improve performance, quality, service, and security, as well as reducing costs. Our first use case starts with a simple business glossary.

## USE CASE #1: BUSINESS GLOSSARY MANAGEMENT AND SEARCH

Many metadata management systems begin with a simple task: create a central location for each business unit to lookup the terms they are using in individual projects. These terms are important not just to orient new people to the project, but also to make sure that business requirements are explained in detail and are consistent across a project and the entire organization.

A business glossary might start with a spreadsheet with two columns: a business term and its definition. MarkLogic can take these spreadsheets and quickly load them into a document store with each term comprising its own mini-document. The MarkLogic Content Pump (mlcp) can automatically split any delimited documents, such as comma separated value (CSV) files, into these documents.

After loading data, the first objective is to build a quick search application. You can use MarkLogic's search function to do this. Start by creating a web form that passes a query off to a service that returns a ranking of the terms that match a query. This function allows you to automatically see ranked matches and provides text "keywords-in-context" highlighting of the keywords directly in the search result text.

MarkLogic's search ranking system also allows any terms that match a keyword to also be ranked higher than matches in a definition. This weighting is called the *word query weight setting*.

The content of a match is also relevant in how the ranking is done. For example, if a keyword like "NoSQL" occurs in the title of a book, there is a higher probability that the book is "about" NoSQL. On the other hand if the keyword "NoSQL" occurs in a footnote of a book we can't make that assumption.

The result is that users get the term they are looking for

right at the top of the search results. It looks and feels like a high-quality search engine, customized to your business users' needs.

Let's say you deploy this application and your users are happy. But now they want to add their own terms, and they want to know who put in a term and who approved it.

The next step is to add a small workflow to each term that shows if this is a new proposed term, a term under review, or a term that has been approved by the project stakeholders. For this you can use MarkLogic's Content Publishing Framework (CPF). CPF allows you to create workflows that manage the state of any term. For example, the term may start out as "New Proposed Term", and then be accepted as a "Draft" term. It can then go through as an "Under Review", "Approved", and finally a "Published" term. At each stage, only an approved list of users can move a business term from one stage to the next. You can prevent users from seeing unpublished terms or allow them to track new terms that are in the approval pipeline.

Just like the "history" metadata in the prior examples, you can track and run reports that show how long terms take to become approved by the team.

As users start to add more terms they want to be able to filter terms based on classification or type. Let's take a classification of the source of the definition. Some term definitions might be taken from legislation, some get their definitions from specific industry standards and some terms are very specific to one team or project. Whatever the classification system, MarkLogic allows you to quickly add filters to your search using a feature called faceted search. Facets are like adding multiple search filters to your search results. Facets are short list of classifications that usually appear in a margin of the search results. Facets and the counts of results by category are maintained for you in MarkLogic's search tools.

Next, your users might note that many terms are related to each other. Some terms might be product categories and some might describe products within that category. In short, your users are leaving the world of flat terms and moving into the area of taxonomy management. By adding an additional data element "broader-term" to these sub-categories you can now create reports that have tree-like structure. This is easy because unlike SQL, MarkLogic's query languages are inherently recursive. They can create a function that displays any root element and then all the sub-elements. For each sub-element the function is called again. Functions with just a few lines of code can create detailed reports of complex depth.

Business glossary terms can also be quickly linked to other terms so that terms can show synonyms and see-also links.

Many MarkLogic customers also like to use web standards for representing their business terms. The Structured Knowledge Organization System (SKOS) is a standard used by MarkLogic and its partners. SKOS allows standard vocabularies to be quickly imported, linked and merged with other terms.

If you type in keywords and do not find the exact term you are looking for, MarkLogic can also use any document as the basis for another query. Finding terms similar to any term can also be done.

What if you are responsible for a specific group of terms? MarkLogic can also be configured to allow you to be notified via e-mail or text if new terms are added or a specify type of term moves from "Under Review" to "Published."

At each step of the way, the core business terminologies are gracefully extended by adding new data elements to whatever terms need them. The application grows and evolves as your requirements change. Your existing programs keep working as the number of elements grows.

If you want to build your own business glossary from scratch, there are third parties that provide business glossary management tools directly on MarkLogic. Many of them allow you to import and export SKOS files directly into and out of their tools. These third parties leverage the power of MarkLogic to provide a great user experience.

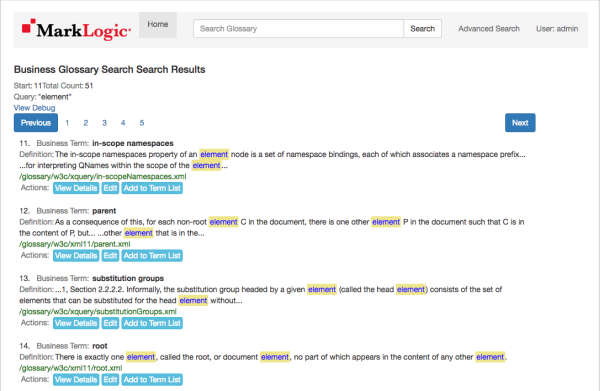



Figure 3. Sample Business Glossary Search Results

The next use case takes a deep dive into one of the more challenging aspects of metadata – the need to handle reference data and provide high-availability real time services from this data.

## USE CASE #2: REAL-TIME REFERENCE DATA LOOKUP

In this case study we will look at how MarkLogic is used to make complex reference data more meaningful and searchable. We will see how MarkLogic's scale-out architecture can be used to provide real-time scalable data enrichment service by enriching numeric codes by adding meaningful labels.

One of the common challenges in legacy COBOL and Mainframe systems was the limited availability of memory. Because of these constraints, developers frequently used short numeric codes to stand in for natural language variables. So when you wanted to record a person's gender, instead of storing the string "FEMALE" in a table, developers would store a numeric code such as "2". Each **application** would then be responsible for using reference data for converting each numeric code into the value in a human readable screen or report.

Despite the availability of low-cost memory, many organizations still prefer to store numeric codes in place of longer human readable strings. For example your US Postal code might be "CA" for California. When healthcare billing is done, an organizational code for "morning sickness" is entered as 11.2. This type of data is known as reference data, because these short-hand codes **reference** common standards. A typical large healthcare organization may have thousands of reference codes that are used in their systems.

Managing enterprise reference data is itself a complex process when codes for states, countries, postal-codes and currency codes may be different across various systems or records. Even within a single system the codes and their meaning may change over time. Some codes, like healthcare charge codes, can have tens of thousands of values with complex rules attached to different codes.

The use of reference codes, in the place of human-readable strings, had some advantages when RAM and disk space was expensive. However, when we build modern document-oriented systems where users want to type in strings into a search engine, we find that requiring users to remember numeric codes has severe usability limitations. Within MarkLogic, the best practice is to store both the numeric codes **and** their natural language search-friendly label in a search-friendly format. We often call this format the harmonized or canonical format. We design canonical formats to support three distinct operations: transactions, search, and analytics. For each numeric "value" in these systems we now store a pair of values and labels. The XML pair might look like this:

```
<PersonGenderValue>2</PersonGenderValue>
<PersonGenderLabel>Female</PersonGenderValue>
```

In order to convert reference data that has numeric codes into the dual value/label format we need some way to insert additional label elements. To do this we will use a simple reference data management framework.

To insert the label, our code will use a lookup function. It will pass in the name of our reference data code (PersonGenderCode) and the current value (2). This function will then return the label. The function call will look like this:

```
ref:label("PersonGenderCode", "2")
```

It will then return the label string "Female".

It is the job of the metadata management system to provide functions where these label lookup functions can run quickly without the overhead required of other

architectures. Let's see how we can implement this in MarkLogic.

When we store reference data, we usually store this data in a simple JSON or XML file. An example of this is the JSON file below.

```
{
   "CodeName":"PersonGenderCode",
   "Items" :
      {"F": "Female",
       "M": "Male",
       "U": "Unknown"
      }
}
```

Figure 4. JSON File Representing Reference Data

We can write a small function that will read in this file, lookup the short-code and return the full string. However, we need to make sure this lookup is fast. Why? Imagine you ingest 1 million new documents per day and each document has 10 codes that need converting. Any slowdown in this service will be costly.

MarkLogic provides two features that can make these lookups happen quickly. The first is the use of server fields. Server fields allow you to put any data structure directly in RAM memory so you don't need to do slow disk-access. The second feature MarkLogic provides is a fast in-memory key-value store called a "map". A map is a key-value store where the key is the short value and the item returned is the label. Using server fields and maps allows you to keep these lookup functions "pegged" into RAM when they are first used, and they remain there until they are no longer needed.

If you are only transforming a few documents occasionally you don't need to use server fields and maps. A simple JavaScript or XQuery function will do the trick and these queries will use MarkLogic's Universal Index so you don't ever need to perform slow scans over reference data to lookup code labels. However, if you need consistent millisecond-range response times, it is nice to know that MarkLogic will efficiently handle these requests.

In summary, MarkLogic provides highly reliable and scalable enterprise-grade document enrichment services. Reference data lookups are just an example of these services.

## USE CASE #3: SEMANTIC SEARCH TO IMPROVE SERVICE AND EXPERIENCE AND LOWER COSTS

In the business glossary case study we showed that MarkLogic elegantly manages taxonomy data. Now we are going to show how these tree-like structures can save your organization both time and money as well as make your organization more flexible.

The recursive structure of XML and JSON make managing these structures easy within MarkLogic. MarkLogic's advanced query languages, XQuery, JavaScript and SPARQL, all easily handle complex recursive network and tree-walking algorithms. The fit is ideal.

Not everyone thinks that taxonomies are true forms of metadata. Some people consider them a form of data organizational structures. However you think of taxonomies, we would like you to consider how valuable these structures can be in helping people quickly find the documents they are looking for, even if the keywords they type in are not directly in the documents! If you think of a taxonomy as a set of subject-trees, with abstract concepts at the top of the tree and highly specialized subjects lower in the tree, then you get the idea of what a subject-heading taxonomy looks like.

Now consider how computers can automatically classify any document. We look at the words in the document and we compare these words to each of the subjects. If words in a document match the subject, then the documents can be thought of as being about that subject. If the subjects have related terms and synonyms, these words can also be stored near the subjects in that tree structure.

Taxonomies can also be thought of as a great place to store a hierarchy of rules for the automatic classification of documents. To classify a document, a document is sent to a service that analyzes all the words in a document. Based on this analysis a document will be classified according to its "distance" to a number of places in a tree.

In the past, automatic document classification systems were slow, expensive and difficult to implement. But with MarkLogic, it becomes easy to integrate document classification systems. We will look at two aspects of making search work better. The first is “keyword expansion” using SPARQL and the second is storing document metadata within documents.

**Keyword expansion** is the process of taking one or more keywords and “expanding” these words to include related terms. For example, if a user types the word “rapid pulse rate” into a search tool, the search system automatically finds a cardiologist near you.

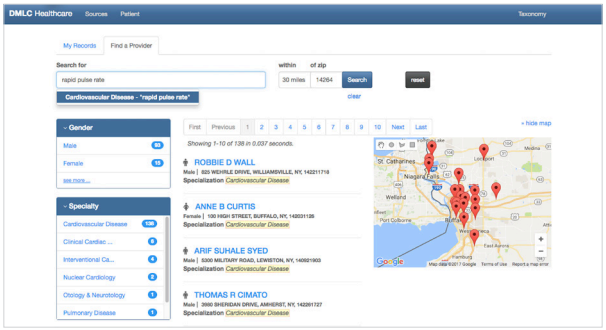



Figure 5. Relating a symptom (rapid pulse rate) to a specialty classification (Cardiovascular Disease) in a search application

MarkLogic has a native ability to store related-words as RDF graphs. We only need to find the initial keyword in the graph and then look around that keyword to find the related words. These words might be linked as alternate labels, synonyms or broader terms and you can increase or decrease the number of related words by adjusting your semantic distance queries.

The second technique is MarkLogic’s ability to add additional **classification** metadata to each document. This metadata can be the result of manual queries that insert additional data into specific documents or it can be the result of a full document classification engine such as the Smartlogic* classification engine. As soon as additional metadata is inserted into a document it is immediately indexed and all words added to the MarkLogic word indexes for that document.

Whatever the process you use to add document classification tags, the additional metadata used in these documents provides an enriched search experience. MarkLogic can combine the expanded keywords and additional metadata to give precise search ranking so the documents you are looking for usually appear in the first page of search results.

Not all organizations are fortunate enough to get consistent, fast and accurate search results. Users spend hours trying to find the right documents, or recreating documents that they cannot find. Yet organizations that depend on getting the right information to the right people know that simple keyword-only matching search alone will not get the right documents to the first page of the search results screen. That is where MarkLogic’s enterprise grade metadata management systems come in.

## MORE INFORMATION

The power of flexible document and graph metadata architectures can make traditional tabular-only systems feel like working with stones and bearskins. Only MarkLogic gives you the security and scalability that you need to quickly put advanced metadata management systems into production.

### BEYOND RELATIONAL

Read this whitepaper to learn why relational databases are ill-suited to handle today’s data challenges.

http://www.marklogic.com/resources/beyond-relational/

### MARKLOGIC OVERVIEW

MarkLogic is the world’s best database for integrating data from silos. Read this overview datasheet to learn about what makes MarkLogic different, including some specific customer examples.

http://www.marklogic.com/resources/marklogic-overview/



MARKLOGIC CORPORATION
999 Skyway Road, Suite 200 San Carlos, CA 94070
+1 650 655 2300 | +1 877 992 8885 | www.marklogic.com | sales@marklogic.com


999 Skyway Road, Suite 200 San Carlos, CA 94070
+1 650 655 2300 | +1 877 992 8885
www.marklogic.com | sales@marklogic.com