The Open Data Model®

DATA GOVERNANCE AND DATA MANAGEMENT WHITE PAPER

OPEN DATA MODEL®

Table of Contents

Table of Figures

Important Notices

Introduction

What is "Data Governance?"

"Data governance is a set of processes that ensures that important data assets are formally managed throughout the enterprise to ensure the data can be trusted and that people can be made accountable for any adverse event that happens because of poor data quality.

*"Data governance is about putting people in charge of fixing and preventing issues with data so tha*t the enterprise can become more efficient. Data governance also describes an evolutionary process for a company, altering the company's way of thinking and setting up the processes to handle information so that it may be utilized by the entire organization. It's about using technology when necessary in many forms to help aid the process. When companies desire, or are required, to gain control of their data, they empower their people to set up processes and get help from technology to do it.

"Data governance is a quality control discipline for assessing, managing, using, improving, monitoring, maintaining, and protecting organizational information. It is a system of decision rights and accountabilities for information-related processes, executed according to agreed-upon models which describe who can take what actions with what information, and when, under what circumstances, using what methods."

The thrust of this Wikipedia definition is that data governance is a process that first and foremost involves assigning responsibility for the desired improvement in data quality. That is, making it a goal for which some individual or group of individuals are responsible and accountable. It is about the introduction of policies and procedures that ensure that high levels of data quality are achieved and maintained.

What does not always come across is that data governance is about the *delivery of a future state*. In this future, the enterprise is using and delivering data that is *fit for purpose* and *correctly represents the real-world construct to which it refers.*

Data governance is *a solution to current problems that solves none of them in the near term.*

Though data governance is suggested as a means by which an enterprise can reach some utopian state, the mechanics of doing so are left largely undefined. Practical steps and timelines are rarely highlighted. Listening to some discussions on data governance, one might come away thinking that assigning responsibilities brings us to nirvana by magic. Metrics that link specific data governance policies and procedures to quantifiable improvements in data quality are rarely if ever seen.

What is "Data Management?"

The Data Management Book of Knowledge (DMBOK) published by DAMA International, the professional organization for those in the data management profession, refers to Data Management (or Data Resource Management) as:

"The development and execution of architectures, policies, practices and procedures that properly manage the full data lifecycle needs of an enterprise."

It also refers to it as:

"The planning, execution, and oversight of policies, practices and projects that acquire, control, protect, deliver and enhance the value of data and information assets."

Clearly, some of the practices associated with data management overlap those of data governance. However, the practice of data management includes an extensive list of associated topics which span the entire process of managing and leveraging data at all levels. A short list of these includes:

- Data Ownership and Data Stewardship
- Data Architecture
- Data Modeling
- Data Quality Management
- Master and Reference Data Management
- Data Warehousing
- Big Data
- Business Intelligence and Analytics
- Metadata Management
- Data Security Management
- Content Management

A well-developed data management program within an organization has the ability to positively affect change around the administration and use of data assets across all levels, departments and lines of business. The benefits of data management includes - improvements in operations management, more effective marketing and sales, better regulation and compliance controls, enhanced security and privacy, reduction of risk across the board, faster application and system development, improved decision making and reporting, sustained business growth, both business and technical alignment, automated and/or streamlined operations, greater collaboration, revenue growth, more consistency across all enterprise processes and numerous others.

Though both deal with data, the definitions of data governance and data management might lead to the conclusion they are carried out somewhat independently of one another. This is largely true but there is one critical aspect of the relationship of data

governance to data management that cannot be overlooked:

The success of one is dependent on the successful implementation of the other.

This dependency creates a boot-strapping problem. There is a minimal level of data management an organization must have in place if they are to be successful in implementing data governance. Invariably it is a level of control that exceeds that which led to the demand for data governance. By the time an organization starts getting serious about data governance, the quality of their data is in a state that will defy efforts to implement it. Without some basic level of control, the chances of successfully implementing a comprehensive data governance program (one that makes a genuine difference) are next to none.

Few organizations truly understand the scale of the problems they face. The sheer volume of data that needs to be bought under control, the diversity of that data and the complexity of the structures that hold it is never fully appreciated. Once they initiate data governance, the size of the task overwhelms them.

This happens because many begin this journey without doing their homework. There is a fundamental calculation that any organisation contemplating a data governance program should make:

$N = A \times T \times C$

Where:

N = Number of data points to be bought under governance
A = Number of legacy applications in the systems portfolio
T = Average number of tables per application (use 135 if you don't have a better number)
C = Average number of columns per table (use 10 if you don't have a better number)

No organisation has any business initiating a data governance program unless they know this most basic of all metrics. As they discover, "N" can get very large, very quickly. For a portfolio of only 20 average sized applications, N will be in excess of 25,000 data points.

In reality, this calculation only hints at the true scale of the problem. To create a permanent solution, you will need to discover and document information about every table. What is its purpose, what is its lineage (has it upstream or downstream dependencies) etc. There are also many things you will need to know about each column. Whether it is a candidate key, whether it part of a primary, alternate or foreign key, its data type and its domain of values, to name but a few pieces of information of which the knowledge is vital for complete control. If you calculate that the number of data points (N) above is 25,000, then taking all these other factors into account means that your total is at least a factor of 10 times larger or 250,000 data points. In our experience, organisations experience quite significant data problems once N reaches 25,000, regardless of whether it represents a small number of large applications or a larger number of smaller applications.

Even though the devil truly is in the detail when it comes to managing data, many organizations believe that a top down process will succeed. But it cannot. You need to understand the minutiae of your data to have an impact on your maintenance burden

because it is at the level of these data points that success and failure are determined. These data points sit at the coal face where your application developers work. It is this level of detail that the coders, database designers and business analysts need. This is a reality that all data governance programs eventually face. It explains why most data governance programs achieve little in the way of permanent change. They don't end up helping the people who most need that help. Data governance programs rarely get defined as failures; rather they just slink into oblivion and the problems remain unsolved. Their proponents discover the hard way that in any enterprise with a reasonable portfolio of legacy systems achieving even a modicum of success is incredibly difficult.

Simply put, enterprises find they *cannot control that which they don't understand.*

The root of the problem is that most organizations have lost the "institutional memory" that once would have provided that crucial understanding.

What do we mean by "Institutional Memory"

Most experienced IT professionals know intuitively what we mean by this term. It is the state the enterprise once enjoyed where they knew everything they needed to know to have complete control of the data created or used by an application system. When we say, *this state has been lost*; it is implicit that at some time it must have existed. When was that? That time was when the application was originally created.

At that time, large numbers of people had to be very familiar with the data structures and their use. They had to design them to meet user needs and those needs themselves had to have been defined. At some point, for the system to have been created, this level of understanding did exist.

The creation of an application is a one-time effort that involves many moving parts. A multitude of people come and go during the process of development. The perfect understanding of the application is spread among all these people. Some level of loss is incurred as one by one the people employed to do basic construction tasks leave the project. *This is the beginning of application entropy.* It may take a while because people on the development team often assume maintenance roles but over time, even these people tend to move elsewhere.

In a perfect world, all of these people would have carefully documented everything. They would have made that information accessible and designed mechanisms to keep it continuously updated as the applications were modified to provide new functionality. They would have made their physical presence unnecessary by providing comprehensive, well organized electronic records that substituted for their absence.

As we discover (literally) at our cost, we live in a world that is far from perfect. Under the pressures that accompany project delivery, documentation is the can that gets kicked down the road. As the originators of a system drift away, their accumulated knowledge leaves with them. The enterprise is left with no substitute knowledge base; with the result being its institutional memory is lost.

Institutional memory is never completely lost

When the developers and designers move on and records that did exist become lost or outdated; many might think that the situation is irrecoverable. *But it is not.*

Though all the people and most records of their work may be gone, they did leave one perfect record behind. That record is the system itself. We don't mean they left behind a perfect system, just that the *record* of what they built, warts and all, *is perfect*. The system itself is a complete, highly detailed record of the result of the choices, good and bad that were made in its development. The system does things. Many of the things it does can be discovered by observing the system and by talking to its end users and those that provide technical support to it.

Generally speaking, the body of knowledge among the user and support community of what a system does will be fragmented. Nobody will have more than a portion of the picture but collectively a great deal of information will exist. Some of that information will be regarding the original system's objectives and design. Many of the original objectives will not have changed. They can be rediscovered. Some goals will have changed and the system will have been altered to reflect them. These changes too will be revealed by examination of the system.

Of particular note, a new body of knowledge will have developed around *what the system does not do and what the system does not do well.*

This body of knowledge will mainly be held by new users, ones who may not have been involved in the original development. These users know what the system does not do that they now wish it could do. They know what it does badly that has resulted in problems that need remediation. These failings of the system are as valuable to us in our quest to recover institutional memory as the records that were never created or maintained.

These users will be particularly aware of data quality problems. Finding the cause of these problems will reveal design and other flaws in the legacy systems that need to be remediated. Applications should be designed to prevent bad data and facilitate the entry of good (standardized) data. Revealing data quality and the causes of it is the first step in remediating it. Only if we have the information about where our problems lie and what caused them can we usefully bring data governance and data standards into play.

Perhaps the most visible of those causes is the inconsistency in data values that arises when what is conceptually the same data occur redundantly across the organisation. Different sets of values for the same thing raise the existential question: "Which version is the truth?" Practically, it causes all sorts of difficulties in reconciliation as inconsistent values commonly occur in the dimensions used in analytical reporting.

Though maintenance may be consuming the bulk of resources, it has become accepted wisdom that that nothing can be done about it. As a consequence, though it may be crippling innovation, reducing maintenances costs is rarely the primary motivation for a data governance initiative. The usual driver is the impact on data quality of redundant and inconsistent data.

What the system does, along with the data it contains is a resource from which we can rebuild our institutional memory. From the system and those closely involved with it, we can discover all of the following:

- What the system does right
- What the system does wrong or could do better
- What the system does not do that we would like it to do
- Where the system allows data quality problems to be created that better design might prevent at source

"Data archaeology"

Discovering what a system does, or should be doing, is only part of what is involved in recovering the institutional memory. The data it creates, reads, updates and deletes in doing what it does is a lot more difficult to uncover. Typically, it involves a process of "data archaeology." By digging through the database structure and content, some idea of the impact a process has on a set of data can be unearthed.

Data archaeology is essentially a random process conducted in response to a specific situation, usually a particular request for change or a fix to a broken process. It is not intended to put together a complete picture of the data resources, just solve specific problems. Data structures can be complex and because it does not take a holistic view, this superficial research frequently fails to uncover the full picture.

Each data archaeology effort is independent of every other effort. The results of each effort are discarded because there is no comprehensive view into which they may be fitted. That is neither an objective nor by-product of the effort. The goal is to determine the impact of a specific change to the application, not to meet any broader goal. Data archaeology as normally practiced is expensive and time consuming (and even more expensive in terms of risks of failure when done poorly).

The fact data archaeology is sporadic and narrowly focused blinds us to the value of the resource it has available to it. Every system will sit upon a database. Most will sit on one of the well-known SQL database management systems (generally referred to as relational databases) such as Microsoft SQL Server, Oracle, DB2, UDB or Sybase.

Few people realize the value of this resource. A database is a highly-structured source of information. Its beauty lies in the fact that a database does not comprise thousands of different kinds of structures but just two - tables and columns. A database may have hundreds or thousands of instances of them but the fact there are only two kinds of structures is critical. Learning what it takes to understand two types of objects is easier than having to understand a much more diverse set of them.

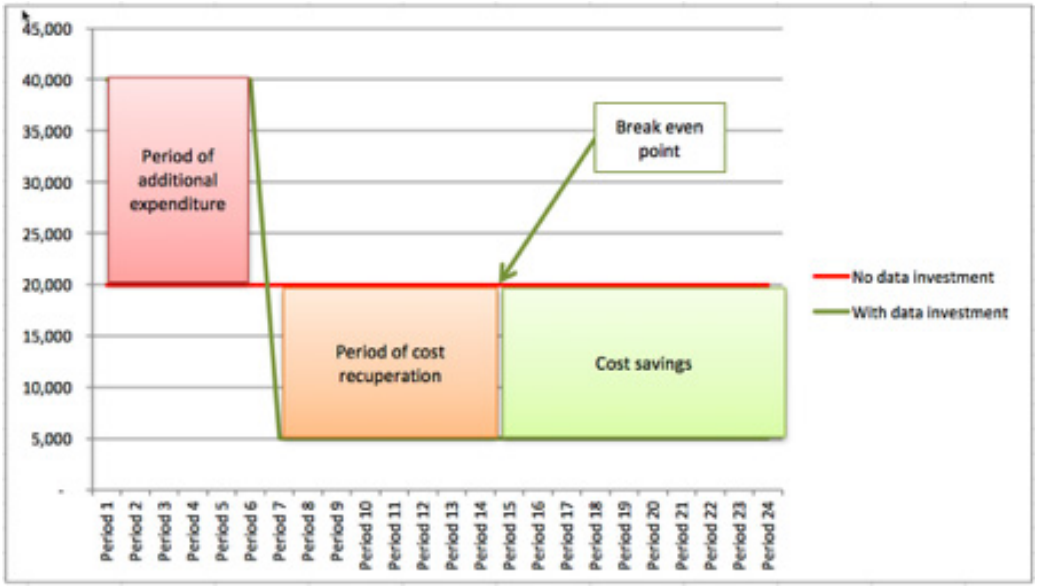
The database can also provide another critical piece of information. Using query tools, *we can see the content of the database*. That is, we can see the actual data that each table and column contains. From a data archaeology point of view, it is not essential that the data being viewed be production data, just that it is representative of

production data such as the data typically found in a UAT (User Acceptance Testing) environment.

When you couple what we can discover from examination of the database and its content with what we can discover from the system and those closely involved with it, we are bound to conclude that the real impediment to recreating our institutional memory in total is not that it is impossible but that *we don't have a well-defined, highly efficient process for doing so.*

What prevents us from attempting it is that it appears that it will be incredibly time consuming. We don't know where to start or where to go and we have no confidence that what we end up with will be worth the effort. Above all else there are always more pressing problems (many of which are, of course, caused by or exacerbated by loss of institutional memory). If we were being clinical about it (which we rarely are) the question of recreating institutional memory is not the feasibility (no one can doubt it is possible) but the very pragmatic one of measuring our "return on investment." At what point is bearing the cost of a holistic approach, solving our problem once and forever, better than bearing the cost of multiple piecemeal data archaeology efforts?

FIGURE 2 – THE OPEN DATA MODEL® RETURN ON INVESTMENT



One way to resolve this conundrum, is to substantially reduce the time and cost of recreating the institutional memory. The faster and easier we make the holistic process, the earlier we get our payoff and the simpler our decision. There is a point along the continuum of faster and easier where the process is "very fast and very easy" and the question of whether to proceed becomes a "no-brainer." This is where the Open Data Model® (ODM) comes in.

Outstanding achievement though it clearly is, the most important benefit the ODM delivers is not enabling the recreation of institutional memory. It is enabling the *cost-effective recreation of it.*

Figure 2 illustrates a very credible real-world scenario. For our benchmark organisation, the one with a portfolio of 20 applications, we can assume the cost of the regular data archaeology component of "lights on" maintenance will be at least $20,000 per month. ($1,000 per application/per month). Data governance programs aim to reduce this monthly cost but rarely achieve this result. Thus, this cost continues forever.

The time for a competent data architect using Open Data Model® (ODM) to recreate the institutional memory of an organization of this scale is around 6 months. Let's assume the cost of the full-time services of a data architect, some cost for collaboration with other users and the subscription for the ODM to be $20,000 per month. Allowing for a period of transition to use of the recreated information, after 7 months our comparable spend is reduced to $5,000 per month. Our break-even is around 15 months out from project start. From that point, not only are our savings accruing but we have identified all our quality issues, mapped our redundant and inconsistent data and built a roadmap for change. Very importantly, the resources that were previously being sucked into the black hole of maintenance are now freed up for redeployment to innovation.

Though we have illustrated the concept with a small model, our practical experience is that the ODM scales even more advantageously. We are well aware that there are many enterprises where the portfolios run into hundreds even thousands of applications. For a variety of reasons, such as upstream and downstream dependencies, maintenance costs increase exponentially with portfolio size. Where the ODM stands out is that the time and cost to deliver its benefits decreases exponentially the more applications are transitioned to it from the legacy system portfolio. The reason is simple. The ODM only needs to model any concept once. From that point it can be used at a fraction of its initial creation cost to identify every redundant occurrence of the concept in the legacy system portfolio. The highest cost is the first application. The least cost is the last. Though it may take a long time to map all of a huge portfolio the ODM pays that investment back from the first and from there on at a faster and faster pace.

The ODM works by taking full advantage of the presence of the highly-structured source of information and content that application systems databases provide. This resource provides the foundation for a systematic approach to reconstructing the lost knowledge base. By "systematic," we mean a set of effective, cost-efficient processes that can be applied repeatedly to the resource. The technology of the ODM enables these processes.

Just as we would argue that Data Governance cannot be established just from the top down, we would agree that neither can it be implemented just from the bottom up. However, we see the establishment of the higher-level controls being so much easier when a certain level of reconstruction of the institutional memory has occurred.

Using Open Data Model® to restore the institutional memory

In our view, data management begins with the regaining of a level of control over the raw legacy system data. Once that has occurred, policies and process can be put in place and people made responsible for continuous improvement.

At the Open Data Model® we believe, in principle, data management is simple.

For complete control of your data, you need to know just four things about every data item:

- What it is
- Where it is
- How it got there
- How it gets used

And have all this knowledge at your fingertips.

The root of all problems that lead enterprises to believe they need data governance is that they do not know these things and they can't conveniently discover them. Some might suggest that if they had this level of control the arguments regarding the need for data governance might never arise. The Open Data Model® makes that argument irrelevant by enabling both.

It begins by enabling the systematic recovery of the institutional memory.

Regaining control starts with the "database schema"

The structure of a database is defined in a database definition language that we can extract and import into the ODM. That definition, which we call the "database schema," gives us the names of the tables, the columns they contain, useful information about them such as the keys they form, the indexes on them, the nature of the data they contain (numeric, alphabetic, etc.) and whether a column can be left empty (null) or must contain a value.

The names in the schema represent the physical implementation of the database. The names may be cryptic or comprised of abbreviations and not easily understood. The real-world data that they contain is frequently unclear. That can be a challenge. Overcoming that challenge is the role of the ODM.

Using a variety of easily learned techniques, the schema and content of the database can be used to tell us almost everything we need to know.

How successful can the Open Data Model® be in "reverse engineering" the knowledge

base? The answer is that we are extraordinarily successful and can demonstrate that this is the case.

What the Open Data Model® recovers is not just what has been lost (the old record) rather; it is a better record, a well-organized, greatly enriched and vastly more accessible record.

Knowledge at your fingertips

In discussing what it takes to provide complete control of data, we emphasized you needed to know what it is, where it is, how it got there and how it gets used. We mention but do not really explain the importance of providing immediate access to that information.

A little bit of introspection will reveal that if we truly mean, "immediate on-line access to all the information we need for control," we are dealing with a very complex cross referencing and indexing problem. To get to some piece of data instantaneously, you need some way to reference that data. You need a way to say what it is you want to see and a way to get to it; ignoring for the moment the fact you may not know what it is called that you want to see.

For a medium sized enterprise we will be dealing with hundreds of thousands of data points. These data points don't fit neatly into some sequential narrative like chapters of a book. Understanding the purpose and content of a single table may require understanding of a dozen tables that it references or is referenced by. Even if not explicitly implemented through key structures, every table will, nevertheless, have at least one relationship with another table. With rare exceptions, it makes no sense for a table to exist in a relational database without at least one connection. The indexing and cross referencing of these relationships is as critical as defining the purpose of each individual piece of data. What you are dealing with in a single database is a web - a matrix of information.

Figure 3 shows what a large data model with all its relationships exposed might look like. In this one (known by the enterprise as the "Death Star), you can see what we mean by a matrix of information:

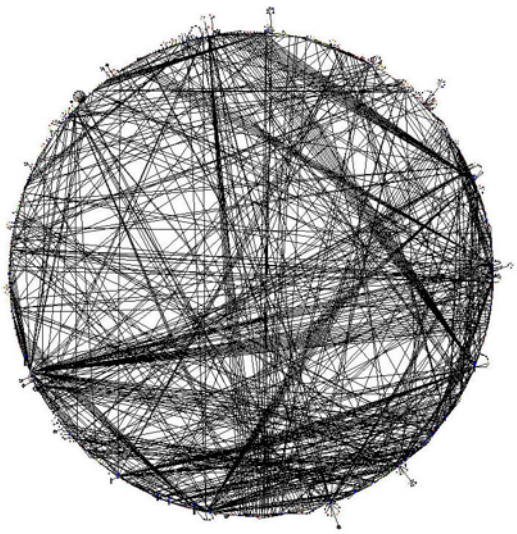


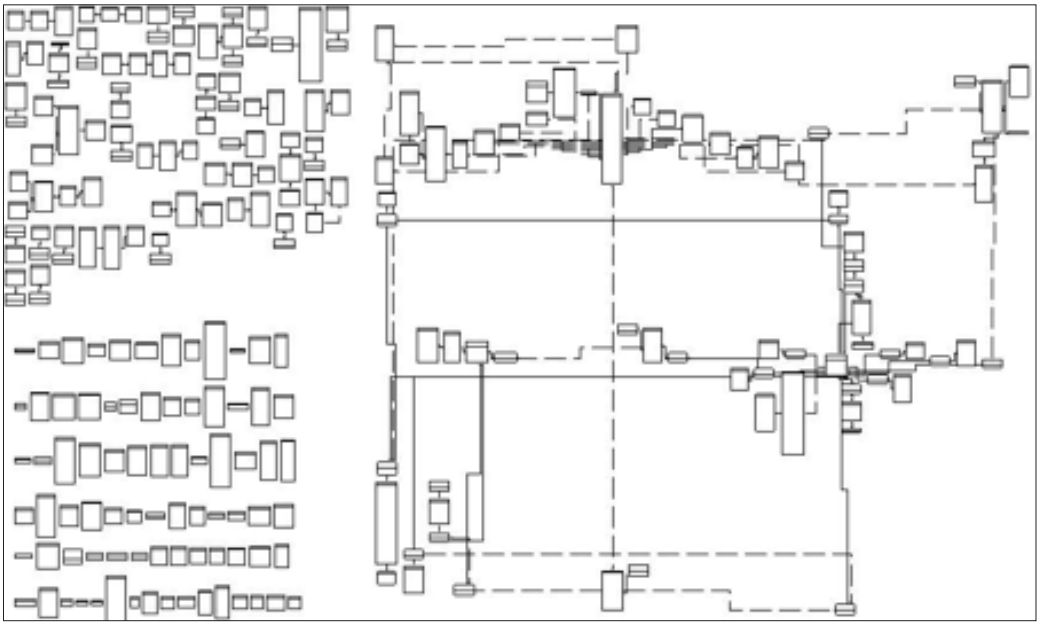
FIGURE 3 – THE DATA CONTENT OF A SINGLE DATABASE (THE "DEATH STAR")

Technically, what we have in Figure 3 is a matrix of metadata. Metadata is information (data) about data. In this case, what we have is data metadata (metadata about data).

Now, some people will say that reverse engineering a database will achieve the same result. If you are following the discussion carefully, you will understand why this is not in fact true.

In Figure 4, a data model very typical of those created by reverse engineering a legacy database is shown:

FIGURE 4 – A DATA MODEL RESULTING FROM REVERSE ENGINEERING



You will note that in this data model, very few relationships are exposed. This happens because in a great many database applications only a portion of the possible foreign key constraints are explicitly declared. In fact, it is not uncommon for none of them to be declared. Foreign key constraints give reverse engineering tools the technical basis to automatically determine what relationships exist. Not having those constraints means that we can't automatically deduce them; it does not mean they do not exist. In fact, a subset of the data in the "Death Star" shown in Figure 3 is of the relationships between entities that in the reverse engineered model in Figure 4 are unconnected.

If the connections between the tables in a reverse engineered database are hidden in code then nothing in the typical data modeling tool is set up to expose the reality of their existence. The Open Data Model® is designed to enable the matrix of interconnections to be derived and the reality to be made visible. The data model shown in Figure 3 was created by the ODM.

But the distinction between what the Open Data Model® builds and anything a data modeling tool can do becomes much more evident when you consider that what you typically want to gain control over is a portfolio that contains tens, hundreds even thousands of applications, bigger and smaller than this.

A data modeling tool is used for the design and maintenance of a single database. In reverse engineering mode, it gives you a picture of a single database and nothing more. Reverse engineering two databases will just give you two stand-alone models. The problems that enterprises face are not the result of design flaws in a single database but problems that span the portfolio of applications. These problems are the inevitable result of years of "silo development."

What is "Silo Development?"

We have explained the loss of institutional memory in terms of what happens in a single application but that explanation is a little misleading. No enterprise needing data governance is in that state just because they don't know about the data used by one application. The demand arises because they have portfolios that contain scores, hundreds or even thousands of applications. Their problem is the cumulative loss of knowledge about the data used by a vast network of interdependent applications developed over decades by hundreds and thousands of individuals who typically worked on just one or just a few of these applications.

Over time, many of these developers moved on. In earlier times, the loss of institutional memory caused by this was relatively slow. Most developers were employees rather than contractors and though they worked on new projects they were still available. This is no longer the case. In the past decade, the disappearance of the knowledge base has been greatly exacerbated by the increased use of contractors and by the phenomenon we know as outsourcing. A little understood consequence of these trends is that knowledge of a system is largely gone the moment the system is installed.

The applications they developed were almost always designed to meet the needs of an individual business group or to perform a very specific, limited set of functions for the business. This is what is known as "silo development." Silos create an environment of individual and disparate systems within an organization.

Forces such as budgets, the need for speed to market or a need to meet a deadline for regulatory compliance also played a part. There is a reason our research tells us that the average database in an organization with a large portfolio contains around 135 tables. It is (or was) approximately the amount of functionality that could be delivered in one year for no more than one million dollars.

The majority of data in any application is there to meet the needs of one of these narrow interests. It sits in what is known as a "data or information silo." Such silos tend to arise naturally in large organizations because each organizational unit has different goals, priorities and responsibilities.

Data silos are a barrier to collaboration, accessibility and efficiency. They impede productivity and negatively impact data integrity. When two or more silos exist for the same data, their contents are likely to differ, creating confusion as to which repository represents the truth. It is data redundancy (the same data in multiple places) compounded by inconsistency (different values for what is logically the same kind of data) that arises among multiple data silos that fuels the push for data governance.

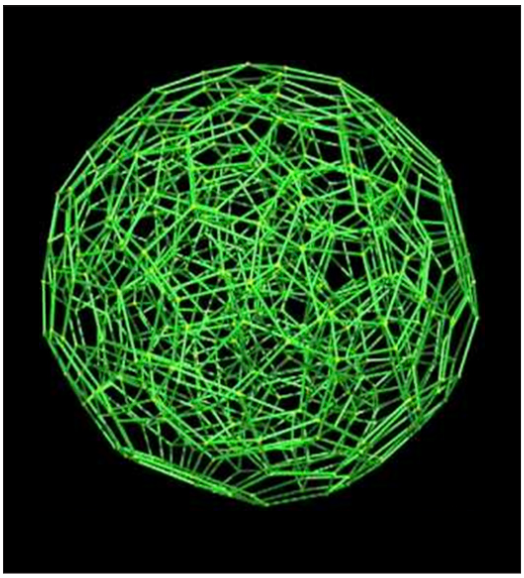
Enterprises know they have high levels of data redundancy. The same data is created and stored over and over again by independent applications. Information about people and places occurs in every application. Reference data tables similarly occur in every application, frequently with their varying data. When used for analytical purposes the inconsistency of measures (a role played by many reference data sets) makes reconciliation of results extremely difficult if not impossible.

The impact of silo development on our indexing system

To meet our goal, our cross-referencing and indexing system must provide a gateway that will take us directly to information about any data item in any one of these applications. That system must take into account that along with unique application specific information, many of these databases will contain the same kind of information (e.g. client related data structures). The cross-referencing and indexing system must instantaneously provide this enterprise wide view to us, telling us every place where information of the same type is redundantly stored. Not just for informational purposes but because this knowledge is critical to one of the benefits we are looking to obtain - that is, the ability to rationalize our data and be sure we have one source of truth.

If you link every legacy system data item to at least one object in the two-dimensional model of the "Death Star" shown in Figure 3 (connecting every legacy system data item that is conceptually of the same kind to the same object in that two-dimensional model) and hold information regarding every relationship between data items in the same legacy system, what you end up with is a multi-dimensional metadata matrix.

FIGURE 5 – VISUALIZATION OF A MULTI-DIMENSIONAL METADATA MATRIX



The image shown in Figure 5 is a fraction of the scale and three-dimensional but will help you visualize what a cross referenced, indexed matrix of interconnected metadata looks like:

Now imagine something as much as a thousand times larger. That is what we need to build as a replacement for our lost institutional memory. It is certainly an elegant looking solution and instantaneous access to information is clearly a huge advantage but this is only one way in which it will be better. The actual information content will be many times more useful.

Why will the new knowledge base be better than the old?

Overall data redundancy is very undesirable. However, for the ultimate purpose, for the purpose of getting it eliminated, it is a very good thing.

Why is that? It sounds contradictory but it is not. The person charged with coming up with the one perfect representation of enterprise data has, in these multiple occurrences, a very valuable source of information. From their perspective, each occurrence offers a different view on what is conceptually the same kind of data. Looking at each

version through the Open Data Model® we are able to use this information to come up with the "canonical" form for the information.

A "single source of truth"

What people rarely articulate but is implicit in what they are trying to achieve is that the goal of data governance is a state where the enterprise is running on a single database.

It is implicit in any statement that they want greater "integration" or to reduce complexity.

It is implicit in any statement that they want "a single source of truth."

It is implicit in the efforts to remove redundancy and inconsistency from their reference data sets. If an enterprise has only one Marital Status table it will have only one set of values for Marital Status throughout the enterprise.

Any drive to have a "Reference Data Hub" serving the entire enterprise reference data needs, is indeed a call for a single copy of each reference data set. Likewise, any call for "Master Data Management" is a call to have a single source of customer or product information.

Very few organizations ever manage to reverse years of silo development and end up with a single database (through at least one Fortune 500 corporation has achieved this goal). Given the technological advances made in database technology since many of these legacy systems were created, this is more than a little surprising. The potential advantages of a single database are huge. They do not accrue just from the elimination of redundant and inconsistent data, hugely valuable though that will be. There are other advantages. No application is an island. Just as departments in the real world of business need to pass information to one another for the business to operate in an integrated way, so do the systems used by those departments. A huge problem with silos is the need for this inter-application communication. That is, a way to pass information from one silo to another that needs to use the information.

An application that needs to have data from another application is said to have an "upstream" dependency on that application. To the application that is sending the data this is a "downstream" dependency. In the convoluted world of application silos, every application has numerous upstream and downstream dependencies.

One of the four things we need to know to have complete control of our data is to know how it got to the place it is being used. Knowing the lineage of our data is a problem compounded by the existence of a complicated network of connections between applications. Applications can be both consumers and suppliers of information. With a single database, this complex network of inter-application communication would be eliminated.

Running a large enterprise on a single database is technically feasible. Modern databases can easily support schemas with thousands of tables. Nevertheless, few people consider making this an explicit goal.

However, what many organizations do is create a model of this "ideal" database.

Large scale data models have been developed in the past. Many arose out of the philosophy of "Information Engineering and in earlier days were known as an "Enterprise Data Model" (EDM). The purpose of the EDM was to help the enterprise gain a better understanding of how the business worked by looking at its data was used across the enterprise.

Some enterprise scale models being developed today serve a slightly different purpose. That purpose is to define data structures to which the enterprise can gradually align its systems. This data model is generally referred to as a "Canonical Data Model" (CDM). The CDM is often referred to as a "future state" data model." The distinction between an EDM and a CDM is by no means clear cut but today "canonical" appears to be the favored terminology for these enterprise scale models.

The Canonical Data Model (CDM)

What is a canonical data model?

Canonical is the adjectival form of the noun "canon" or rule. Canonical means conforming to well-established patterns or rules. In the context of data governance, a "Canonical Data Model" (CDM) is an "authoritative" data model.

The CDM represents the inherent nature and characteristics of the data, its structure and how the different parts relate to each other. It defines the things the enterprise stores data about (entities), the characteristics of those things (attributes) and the relationships between these things. The CDM has no replicated data. It is independent of any application, any database management system or computer hardware. It is a minimally redundant, process independent, data model.

The Canonical Data Model is the best representation of the enterprise data, one that has structural integrity, is complete, accurate and resolves all the conflicts and inconsistencies in the structures currently holding the comparable data. It is inherently the most stable data structure available. It is the most capable of being extended in the future with the minimum of alterations needed solely to accept the additions.

Building the canonical data model

Just in the scale of its coverage, building a CDM is a very demanding task. The typical enterprise will have a network of inter-application data transport paths, scores, hundreds or thousands of interacting legacy systems and from tens of thousands to millions of data items.

Traditionally, building a CDM is a labor-intensive task. Rather than using the schema and content of a legacy system as a source, the typical CDM project involves countless meetings with the users of the legacy systems. The developers of the CDM try to determine the data that the future systems will need through discussions with the users of existing systems.

The development process will involve reading whatever documentation, however dated, that does exist regarding the application. It may include consideration of the data on forms and on-line interfaces. The approach may involve looking at data models of the legacy systems but in the absence of technology that the Open Data Model® provides, this examination is neither systematic not core to the process. This haphazard approach to gaining information is inefficient and ineffective.

One of the critical problems is that the process of creating the CDM does not of itself deliver value. The value of the CDM is seen in its completion, not in any interim result it can provide. The lack of quantifiable benefits delivered in timely manner ultimately gets questioned. Most CDM projects run out of steam and fold their tents and fade away.

If it does not die before reaching that point, the delivery of the future state model raises a whole new set of issues. Most importantly, now they have created it, how does the organization move to the CDM? The enterprise has defined its current state and future state but the current state is a multitude of isolated towns and villages scattered across the data landscape. In their inter-application communications paths, we have backroads connecting one to another but no roadmap exists connecting each of these towns and villages to the metropolis.

Further, each effort to create a CDM starts from nothing. Though patterns do exist for common data structures, few people take advantage of them and if they do, it will be to follow the logic rather than replicating them physically. At the other end of the scale, there are enterprise scale data models (industry data models). If you are one of the rare organizations for which such a model is a good fit and you can afford it, such models are hugely beneficial. The Open Data Model® is for everyone else.

Through a systematic, highly efficient process it enables rapid development of a CDM that delivers value from the moment the first artifacts are created. It immediately provides “where used” information that helps reduce maintenance costs and risks. From the beginning, it provides a roadmap to the future state that is tangible and practical. Best of all, you do not start with nothing.

The Open Data Model® is a federated model of public and private models. Any organization building a CDM will build it in a private domain accessible only to designated members (e.g. their own staff and consultants). However, in the public domain all members have access to a range of models created by Special Interest Groups (SIGs).

The Open Data Model® is designed to facilitate collaboration between members. That cooperation may be within their private world or in a public SIG. Through this teamwork, models get developed that provide solutions to a wide range of common business problems. Any component of any model in the public domain may be “cloned” (copied) by any other member into their own private model. A component may be a single entity, the full set of entities it is related to and those relationships or a complete model.

Nobody in the Open Data Model® needs to spend time designing a good model to handle people and places. Such models have long ago been developed and proven. Collaborating and contributing in the public domain is encouraged. Collaboration gives us the benefits of the collective experience and expertise of the membership. Without it, we rely on just our individual capabilities. If everyone makes small contributions and everyone has access to the cumulative efforts of their peers, the result will be that everyone gets far more than they give.

The goal in building a CDM in the Open Data Model® is not to mimic any given set of structures implemented in any of the legacy systems but to use information gained from them all to build a model that captures their real-world nature. The CDM equivalent will be a representation of the same concept. However, it may be structurally different to some or all of the legacy system implementations if that is deemed necessary to provide a better solution.

The process looks at all the systems that form the enterprise landscape bringing to the fore integration issues and different points of view. These get teased out using the collaboration facilities of the Open Data Model® to share knowledge and to reach consensus through debate. Besides being very efficient, the added benefit that the Open Data Model® provides is an audit trail of all the reasoning behind any design decision.

Of great importance is examination of all the problem areas within the legacy systems. It is common to find that many of these issues resulted from misunderstandings of the enterprise reality and/or assumptions about the nature of the business and its data that turned out to be incorrect. There all also issues revealed that can simply be attributed to poor design. They do not get replicated in the CDM.

The exercise results in a design that shows how the data elements should have been structured and should be structured in the future if we are to eliminate these problems.

It is the role as guide to the potential remediation of legacy system problems that the CDM is seen as providing its greatest value.

Being the definitive model of how things should be a canonical data model will frequently be called a “future state” model.

Though the reason for this might seem obvious, the term is a little bit of a misnomer. It is true that the CDM represents an ideal state for the enterprise data but technically it is a model of the current state. The CDM does not contain any information about new functionality, unless catering for new requirements forms part of the mandate and thus information pertaining to those new requirements is provided.

It does define a future state to the extent that the enterprise could eliminate many errors in their existing systems by re-engineering them into this improved data structure and it is presumed the enterprise does have a desire to do so.

Rather than use the term “future state model” it is valuable to think of the CDM as a “picture of reality.” If well done, any expert in the business should, on examining it, see that it captures “the truth” about the business and in using it, they will be able to understand the true nature of the business. Only in this way can we meet one of the

stated goals of data governance. That the enterprise is using and delivering data that correctly represents the real-world constructs to which it refers

That the canonical data model is a picture of reality is a core concept of any CDM built in the ODM. Why do we emphasis the concept? Because if your model truly captures the intrinsic nature of the data in the enterprise, it solves a problem we always face with building anything… where do we start?

Where do you start?

If the canonical data model correctly captures the truth then it is axiomatic that anything added to it that also accurately capture the truth, will fit into it just as it does in the real world.

A useful analogy for the process of building the CDM is that it is like "paining by numbers." Figure 6 is an example for anyone too young to know what "paining by numbers" looks like. Each number is a key to the color to be used in that area. If you use the correct color in each numbered area it does not matter where you start, the complete picture will eventually emerge.

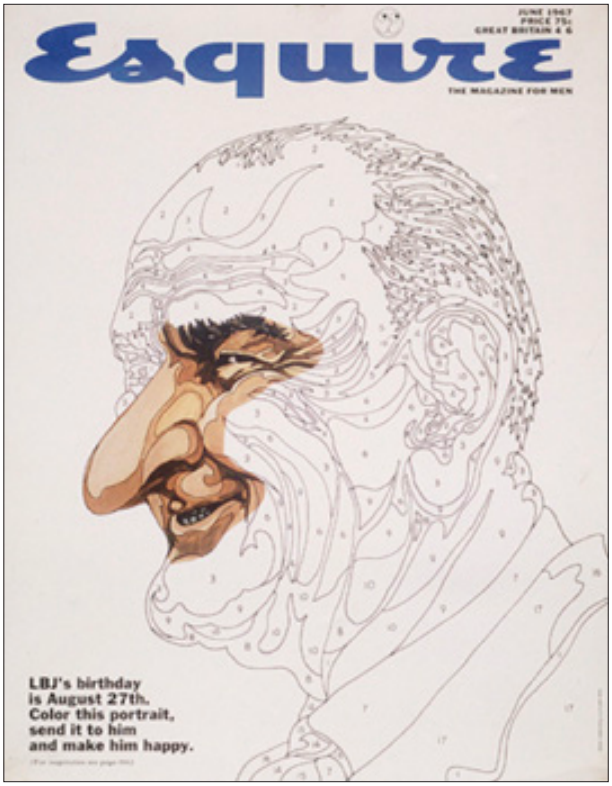


FIGURE 6 – PAINTING BY NUMBERS

In practice, the Open Data Model® functions exactly as we say it should in this analogy. It really does not matter where you begin. For efficiency, we do suggest users focus on an area at a time and try to build a relatively complete picture of that area (rather as a portion of the portrait shown in Figure 6 has been completed).

We encourage anyone involved in the building of a CDM to understand this concept. There is no cause for anxiety. In practice, if you do the work competently, it will work out exactly as advertised. We do have suggestions on how to most efficiently build the model but where you start and finish is a question of priorities. Start where you feel you have good information and go wherever that leads you. To paraphrase Dirk Gently, "holistic detective," rely on the fundamental interconnectedness of all things.

The role of the Canonical Data Model in the Open Data Model®

The maintenance burden

Data governance is high on the agenda of so many enterprises today not because they have "got religion" but because they are experiencing significant costs in its absence. The underlying problem data governance initiatives are trying to solve is what is known as the "maintenance burden." The maintenance burden is what we call the huge volume of resources that are dedicated to enabling "business as usual." The cost of what is frequently referred to as "keeping the lights on." Gartner Group estimates that maintenance consumes 50% of IT budgets and a whopping 80% of resources.

The phrase "legacy systems" might create the impression that these are systems that just run day after day without needing any attention. This could not be further from the truth. Though they may not change much, all legacy systems do experience some constant change. The collective cost of implementing these changes can run into millions of dollars.

The costs and risks are not confined to the system being changed. In the world of interconnected information silos there is also the cost of ensuring that they do not have unintended consequences on applications that are consumers of data produced by the system being fixed ("downstream" applications).

What we are trying to deal with in data governance are factors that contribute to this maintenance burden:

- Poor data quality
- Redundant data
- Inconsistent data

Why is it so difficult to implement change? Because without help of technology like the ODM, it takes a lot of time to evaluate the impact on legacy systems and plan a path forward. Taking this time is essential because poorly planned changes pose high risks to "fragile" but vital systems. It is work that requires proficiency and is thus expensive. But the alternative for the enterprise is equally bad. The choice is to pay at the front end or pay more at the back end.

The root cause of these high costs is the absence of readily available, in-depth knowledge of the legacy systems i.e. the loss of "institutional memory" that we have been discussing.

The maintenance burden is not a new problem but it is one that is getting steadily worse. Like any longstanding problem, technology has been used to try and resolve it.

World-wide there are billions of dollars to be saved by reducing the cost and risks of maintenance efforts. It has not proven an area that is easy to address but for a long time specialized solitons called “Metadata Repositories” have been providing this capability to a small number of enterprises.

Metadata repositories are not to be confused with repositories of metadata. Many IT tools need “internal metadata” to work. Metadata repositories import this metadata – which to them is “external metadata” and use it, not to replicate the functionality of these tools but to produce “Impact Reports.”

Metadata repositories were founded in the belief you could “boil the ocean.” That is, cross reference all your data and processes. The idea is that if you know everything about everything, you can see the impact on data of any process change and the impact on processes of any data change. With fragmented repositories of poor quality meta-data, enterprises struggle to obtain any such unified view.

Vendors built a range of tools called “adapters” to enable import of everything from code libraries to data models. On the data side, they were used to import models and schemas from data modeling tools and database management systems and merged to create a universal data dictionary.

When we speak of metadata repositories we are in rarified air, a market in which there was never more than a few vendors selling very expensive products.

The traditional model of a metadata repository is a complex point-to-point network in which every object is linked to every instance of every related object.

Figure 7 shows such a model:

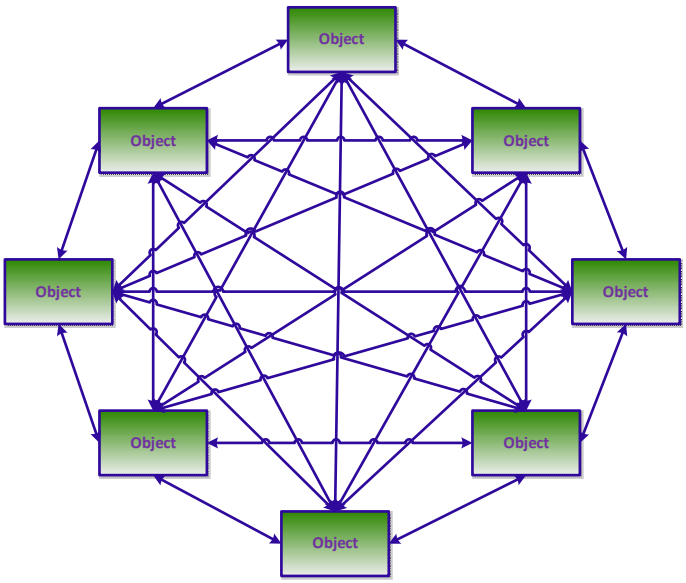


FIGURE 7 – POINT-TO-POINT NETWORK MODEL

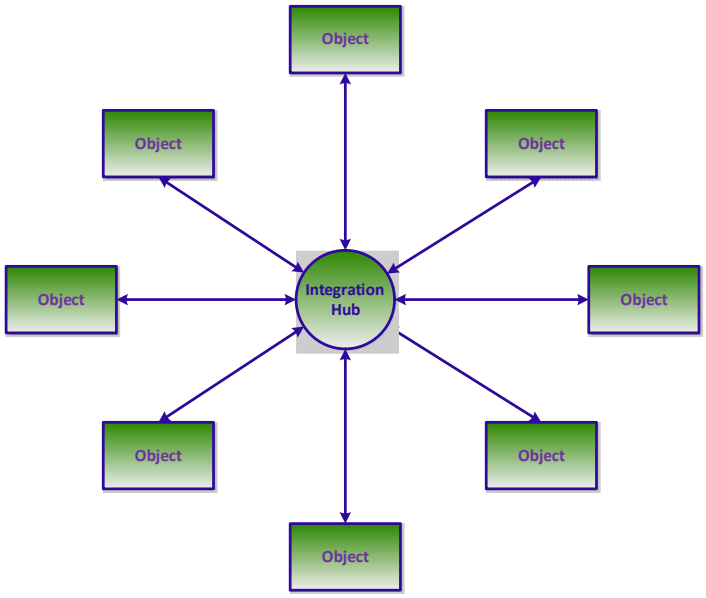
From object “CSTMR” (of type “Table”) in one database the point-to-point model is directly linked to object “tCust” (also of type “Table”) in another database.

One of the questions people will ask, if such functionality is available, then why is it not in more widespread use? The complexity of the network model made it difficult to scale and the resources needed to maintain it proved a barrier to widespread use. It turns out that boiling the ocean takes a lot of work and is very expensive! However, it should be remembered that in these repositories, the ocean we are concerned with comprises processes and data, a collection of data points exponentially bigger than that of processes or data alone.

In an attempt to resolve the scaling issue, an alternate "hub and spoke" model was suggested in which related instances would be linked via a central semantic model.

Figure 8 shows such a model:

FIGURE 8 – HUB AND SPOKE MODEL



Semantic models are based on principles of ontology (meaning) and taxonomy (classification). In this model, the logical entity "Customer" in hub is separately linked to "CSTMR" and "tCust."

Data models are a form of a semantic model. However, they are typically developed as a step leading to the creation of physical database schemas. They are not designed to play the role of an integration hub. In that role, they and the tools that are used to create them do have their limitations. In particular, they are limited in their ability to cater for "multiple inheritance."

Multiple inheritance is a property common to object databases but not easily implemented in relational databases. It allows you to be a member of (inherit from) multiple taxonomy. This constraint resulted in specialized ontology modeling tools like "OWL" being proposed for the development of the semantic hub. Versus widely used data modeling methods, this specialization greatly limited participation.

A breakthrough made in the Open Data Model® is a very advanced data modeling capability that allows a relational database to behave like an object one. The Open Data

Model® supports multiple inheritance. A CDM created in the Open Data Model® follows a semantic modeling paradigm that allows an object to belong to multiple classes. This makes it the perfect option for a semantic model to sit at the center of a hub and spoke style metadata repository.

Why is having the ability to create taxonomies at will so important? There are a number of reasons but one that will resonate with database designers has to do with the practice of "subtyping." It is a common in the logical model to use subtyping to demonstrate the true nature of things. For example, to demonstrate that a Party (supertype) may be either a Person or Organization (subtypes).

The problem is that implementing a supertype/subtype structure in a physical model is difficult. The content of subtypes may be "promoted" to the supertype (Party holds columns for attributes peculiar to both Person and Organization, such as a Person's birth date). You now have data quality issues as none of the columns can be mandatory. Alternately, super types may be demoted (Person and Organization exist separately, with just their attributes). You now have the difficulty of treating them as having common relationships (such as to addresses). You also have the issue that in the physical implementation you cannot have things belong to more than one supertype (that is, there cannot be another taxonomy of which Party is the Supertype).

There are good practices governing when one should promote subtypes, demote supertypes or leave them independent. However, in a legacy system you inherit design choices made years ago. Taxonomies are the equivalent of supertype/subtype structures. Having a CDM that allows taxonomies at will allows you to say that this table (of mixed Person and Organization information) is really logically a supertype and these components belong to Person; these belong to Organization and these to both. Now you are in position to look at how in future you might better handle the reality it illustrates.

The Open Data Model® Data Metadata Repository has at is core a Canonical Data Model. It has functionality that includes an adapter for import of legacy system database schemas and allows new versions (updates) of schemas to be imported and compared to previous versions or compared to an unrelated database schema. It also includes a Business Glossary.

The repository extends the hub and spoke model to allow multiple hubs. Hubs can be swapped in and out as appropriate. The CDM may be used as a hub and "mapped" to any number of legacy schemas. Alternately, the Business Glossary may be used and "mapped" to the CDM or any number of legacy systems. In the near future, the Open Data Model® will allow the creation of a master hub and for this to be mapped to multiple CDMs (that in turn are mapped to legacy systems). This would allow company could have a view across diverse divisions. It would allow Government to have a view across all ministries or agencies.

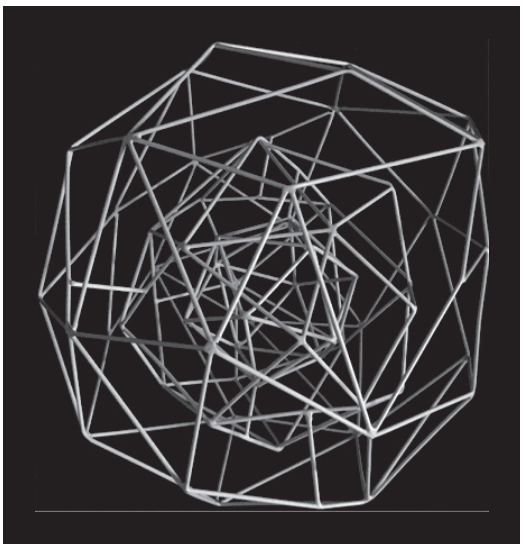
The Open Data Model® enables the cross-referencing and indexing that you need to support a "multi-dimensional metadata matrix." At the heart of the matrix is a canonical model developed using the data modeling functionality of the ODM. The CDM is linked to every object in every legacy system giving us an enterprise wide view of the data across all those applications.

The Open Data Model® lowers the cost and risk of maintenance by providing immediate access to complete and current information. It enables enterprises to restore their "institutional memory" and gives then a roadmap to migrate from the current state to the future state.

Cross referencing and indexing the data resources

Because most people are accustomed to indexes of the static kind such as those found in documents (like this one), it is probably not immediately obvious how the CDM works as the hub of a data metadata repository. Fundamentally, it works because it is not a document; it is a cloud based application that is intended to be used interactively.

FIGURE 9 – VISUALIZATION OF A HUB IN A MULTI-DIMENSIONAL METADATA MATRIX



In Figure 5, we used the image of a dense three-dimensional model to give you some sense of the scale of the matrix. Though it succeeds in illustrating this, it is difficult in Figure 5 to see the part played by the integration hub in that matrix.

Figure 9 is of a similar model but better illustrates how the Business Glossary, CDM and legacy systems are interconnected. In the center is the Business Glossary. Surrounding and connected to it is the Canonical Data Model. Surrounding and connected to it are the legacy systems.

Neither Figure 5 nor Figure 9 should be taken too literally. They are to help you get a sense of the sophistication of the Open Data Model® metadata matrix.

Every table and every column in every legacy system is connected to their equivalent in the hub. If tables in separate legacy systems represent the same concept, they will all be linked to the same equivalent. Because silo development results in high levels of redundancy, the number of objects in the hub will be far less than the sum of all the objects in all the legacy systems.

To give you some sense of how this works in practice, take another look at the data model shown in Figure 4. What you see is in fact a Canonical Data Model. This one forms the hub of an Open Data Model® created metadata repository for a large government enterprise. It contains about 850 entities to which over 2,500 tables are completely mapped. We have gathered interesting metrics on the number of entities a CDM needs to contain to cover the information resources of an enterprise. One that

has always held true and points to the prevalence of redundancy in the silo world is this:

Though the size of the CDM increases as you connect each legacy system, the ratio of objects in the hub to objects in the legacy systems continuously declines.

From the hub, you can reach any object that is of interest and from any object of interest you can discover anything to which it is connected. You can move back and forth from the hub to legacy system objects, then, by finding its equivalent in the hub, find every object that represents the same concept within a single legacy system or across all legacy systems.

How the Open Data Model® works goes back to the earlier statement, the canonical data model is a model of the reality of your enterprise.

Through systematic examination of the legacy systems coupled with the breakdown and reconstruction of each concept they embody, we enable you to build a CDM that is a fully normalized representation of each of those concepts. It is the future state to which all the structures in the legacy systems should ultimately be aligned. In the CDM, every data item lives where it belongs and every data item belongs in just one place.

How we create the Canonical Data Model

In the beginning...

In the beginning, your private domain in the Open Data Model® will contain no Canonical Data Model.

The CDM is designed to hold the best representation of every concept that exists in your legacy systems. It is a model of the one database that could theoretically be used to support all the applications that use the enterprise data. The CDM will form the hub of the metadata repository and be a permanent record of all the data in the enterprise. A record that is always current. The Open Data Model® processes rebuild the institutional memory. The Canonical Data Model holds it and ensures that it is never again lost.

The building of the CDM and cross-referencing it to the legacy system are activities that proceed in an iterative manner. The first step in ensuring that the CDM we build will help resolve our governance problems is to ensure that every object created in it follows high standards of documentation. The Open Data Model® makes this as easy as possible by embedding standards in the modeling process.

Standards are built in, they are not optional and they cannot be bypassed. We insist on documentation at the time of creation rather than hoping it will be added at some

future date. The logic behind insisting on the entry of good documentation at the time objects are created in the CDM is the proposition that if you cannot properly describe what it is you are creating, you have no business building it. Through a workflow that encourages peer review, we further ensure a uniformly high standard of documentation.

The documentation is important because it is going to serve a dual role in the CDM.

The first role of the documentation is to tell us what the object in the CDM is and how it is going to be used. Using the Open Data Modeler® (our design tool) it is very easy to document each object in the CDM as it is created. The Open Data Modeler® allows comprehensive explanations, including links to data sources. In addition, it allows classification of the data in a variety of standard and user defined ways that bring order to it.

The second role the documentation in the CDM plays is to tell us what the equivalent in every legacy system is and what the legacy system object is used for. We are not going to do that by asking you to add a screed of documentation about every table and column in the imported legacy system schema. We want to save you that huge effort. Recovering the institutional memory is not about laboriously doing the job the developers should have done when they created the application. You document the majority of legacy system objects by linking the object in the CDM to its equivalent in every legacy system in which it occurs.

By linking, we literally mean drawing a line between the two objects (mapping one to the other). The simple exercise of drawing that line triggers a number of indexing and cross referencing activities that result in creation of a repository of metadata that is superior in every way to the old institutional memory. Worthy of note is that if this documentation and linking is done at the time of creation of new applications these systems will never have the maintenance issues that bedevil legacy systems. Resource requirements for maintenance will be reduced. The resources that are currently eaten up by the maintenance burden can be redeployed to the provision of innovative new solutions.

Once you get used to the mapping process and its results you will realize that 95% of the time just knowing an object in the CDM and an object in the legacy system are equivalent is enough. In general, what is needed to transform tables and columns from their current state to their future state will be obvious. All you need to add to the documentation of the object in the CDM is anything that is noteworthy about the object in the legacy system. It may be something that is currently problematic in the legacy system.

If the entity in the CDM is classified as a reference data entity, we recommend you query the legacy system and paste into the exposition tab of the CDM entity the set of values you find in each equivalent in each legacy system (there is plenty of room in the CDM). It is not just that it is good practice to document the CDM entity with examples. You will find that capturing the values used in all legacy system equivalent tables will enable you to build up a picture of how the same concepts are implemented across the application portfolio. Most importantly the practice will reveal inconsistencies in the reference data sets to the person responsible for its governance.

The process is dependent on the existence of the CDM. To state the obvious, if you don't have one then one must be created and populated. The first step in doing this is to create

the first objects in it. Begin by building one, or a small number of related structures based on something seen in the legacy system. In doing so, bear in mind this caveat - what you are building describes a reality that is a permanent and integral part of the enterprise. That is, the model does not blindly mimic what is found in the legacy system.

Ask these questions:

- What concept does this object in the legacy system represent?
- What is the best way to represent that concept in the Canonical Data Model?

Though we are confident it ultimately does not really matter where you start, there are certain choices and approaches that will affect the efficiency of your development. The first application is the most important first choice. Choose one from the subset of the applications that deal with core processes. Of those, decide which is best documented and best designed. What you are looking to use as a starting point is an application that is both important and easy to understand.

Why is this choice important? One of the things that you want to remove is redundancy. You want to replace redundant occurrences with a single version of the truth. The CDM will only contain that one version. It will be cross-referenced to its equivalents in all applications. You can take advantage of the existence of redundancy. If you create the most obviously replicated structures first, then they are there for cross referencing to every application in which they occur. Virtually every application will contain information about certain concepts such as people and places.

Make life easy for yourself by using as the source of inspiration an application that is easy to understand. Either one that is recently developed and well documented, or one where the schema demonstrates the developers followed good practices, used standards and were consistent in their design approach.

Make it even easier by drawing on the work already done by others in developing good models for data common to all enterprises. Before you begin digging into your selected legacy application, build up a base CDM by cloning the content of any useful model developed by a Special Interest Group. Cloning is “additive.” You can bring components of multiple public models into your model to build them up. One of the special benefits of the Open Data Modeler® is that models created using it are guaranteed to form an integrated model regardless of their source.

As you reverse engineer more and more applications, you will gain an interesting insight into human behavior, or at least the behavior of people who design databases. Even though every database schema is different, within a schema the designer is almost always internally consistent. The key is to find the “Rosetta Stone” that explains the designer’s idiosyncrasies.

In some cases, the CDM will capture a data structure almost in the identical form in which it is currently captured in some legacy system. In other cases, the CDM design will be better. For example, it may remediate problems that have been identified in the legacy systems. Some structures will be abstractions such as the concept of Party and Party Role.

The CDM "becomes the truth"

One of the concepts we emphasized about the CDM is that it is a model of reality. It represents the true nature of the enterprise of which it is a model. We should perhaps refine that definition to one that reads "it ultimately represents the true nature of the enterprise of which it is a model.

We add this qualifier because we are regularly told of situations where following a CDM reputedly resulted in a flawed design. Such comments illustrate the person making them does not truly understand the concept not that the concept is flawed.

By definition a CDM cannot be wrong!

If it does not accurately represent the enterprise then it is not "canonical" or more likely, "not yet canonical." It has yet to reach that state.

Reaching the true canonical state is not something that is easily achieved at first attempt. If that is the expectation then that is where the problem lies. Much of a CDM will be modelled correctly from the beginning. However, as more systems are reverse engineered, facts will periodically emerge that contradict information previously assumed to be true. It is an iterative process in which the CDM becomes closer and closer to the ideal state we are looking for it to achieve. The CDM becomes increasingly stable with each of these iterations. Nowhere is the aphorism "the perfect is the enemy of the good" more appropriate than in the creation of a CDM. Even an imperfect early stage CDM created in the Open Data Model® will be hands-down better than any model created in ad-hoc fashion if the goal is a system spanning solution to a broad range of problems.

Mapping the legacy systems

Available to the user of the ODM Data Governance functions is the ability to use a browser to find any object in the CDM. From any object in the CDM the browser allows you to find all its equivalents in all the legacy systems. Alternatively, the user may go to a legacy system to answer the question, "What is this data item?" From any table or column in the imported legacy system schema the browser allows you to find its equivalent in the CDM (and from there every other legacy system equivalent in a never-ending series of connections).

The use of the Open Data Model® metadata repository as a means of managing legacy data is based on the concept of equivalence. What you end up with in the CDM is a structure that is a conceptual equivalent of a structure that exists in a legacy system. The way in which the concept is modeled in the CDM may be similar or it may be very different to the way it is physically implemented in the legacy system.

What we allow you to do is create a connection between the two that tells you that these two things are equivalent. The one in the Open Data Model® CDM will be well explained. It is possible that just making the connection will be enough to enable a person starting from the legacy system to say "this is what this object in my legacy

system is about." This object in my legacy system is the equivalent of this object in the CDM. As you create more of these connections you are increasingly able to also do the reverse. That is, to start from an object in the CDM and use it to see all its equivalents in all the legacy systems. This is what is called a "where used" map. Where in my legacy systems is this object used?

The process by which we achieve these results is by creating "Mappings."

Creating Mappings

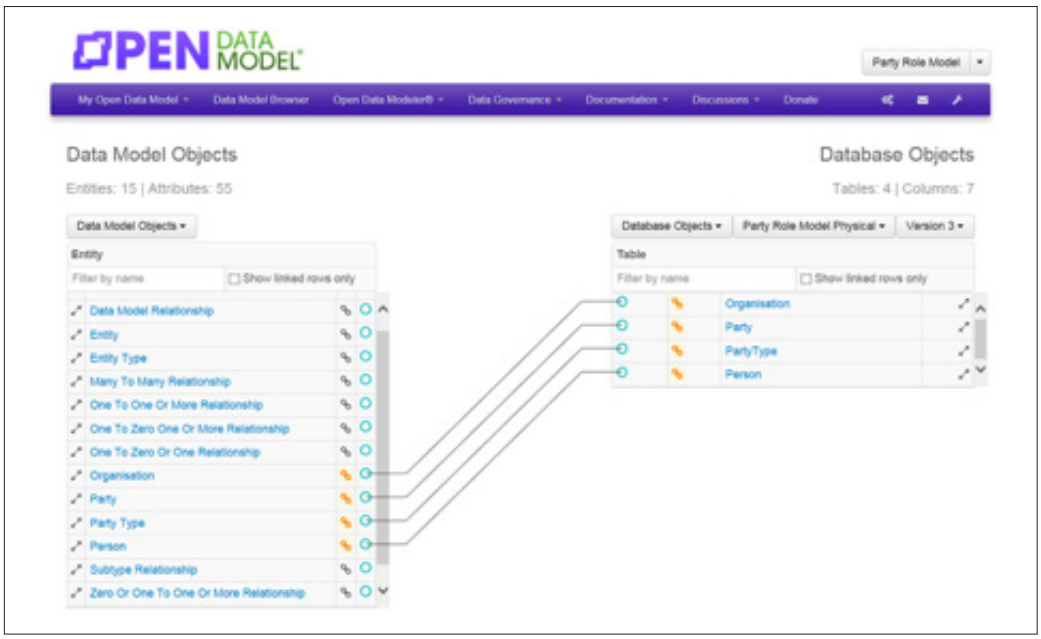


FIGURE 10 – CDM ENTITIES "MAPPED" TO TABLES

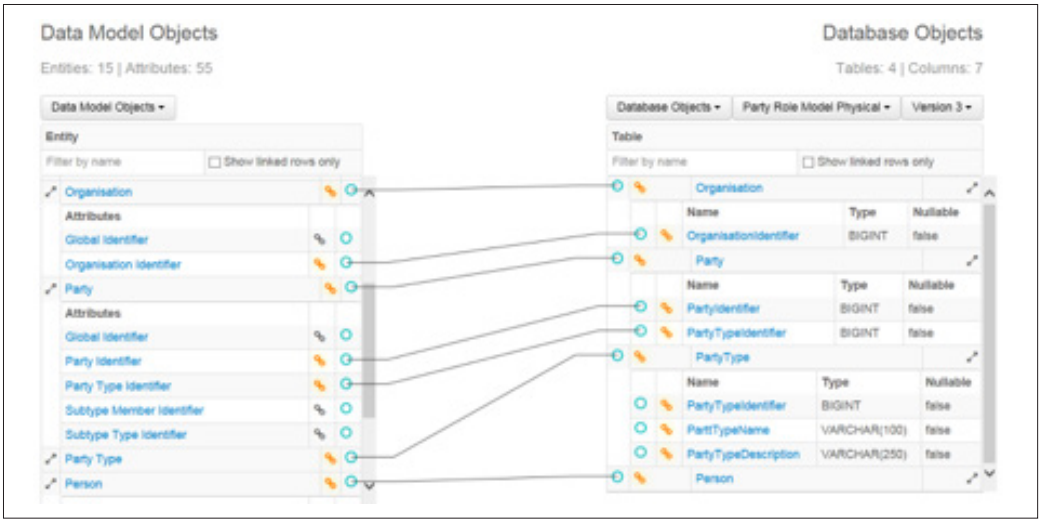
Figure 10 shows how objects in the CDM are linked to objects in a database schema:

On the left of the workspace you have the CDM. On the right, you have links to database objects in a selected legacy system. The process of creating these mappings is simple. Just place your cursor in one of the roundels on the left or right and drag your cursor across to a roundel on the other side (which will change color to indicate you are connected).

Figure 11 shows how the CDM is linked to objects in a legacy system but this time we have used the expander buttons to show the attributes of some the entities on the left and the columns of some of the tables on the right:

It is normal for entities to map to tables and attributes to map to columns but mappings can get much more complex. We don't always create equivalents map one-to-one as neatly as in Figure 11.

FIGURE 11 – CDM ENTITIES AND ATTRIBUTES "MAPPED" TO A TABLES AND COLUMNS



Creating the metadata versus using it

Creating the Open Data Model® CDM and populating it with the legacy system data through the mapping process is a job that requires skill and persistence. The person that will do the task is going to be someone with experience and expertise as a data modeler or data architect. They will be greatly helped in getting the job done quickly if a range of people, particularly business analysts and those familiar with the system processes are available. Their collaboration makes the task proceed much faster.

Collaboration in the Open Data Model® is much more than just getting input from others on questions that are raised, discussed, resolved and forgotten. Discussions are permanently associated with the object that is the subject of discussion. If an object was the subject of discussion, the existence of that discussion is highlighted. Users will be able to see the thread of any discussion. Sometimes it is not enough to know what something is, you want to know why it is that way. The permanent linking of every object to any discussion about it enables that question to be answered no matter how much time passes.

Creating the metadata is a one-time effort. Progress follows its own variant of the 80/20 rule (the "Law of the Vital Few"). Very soon in the process, the growth of the CDM slows down and the task is increasingly about making the connections to the legacy data. Once all important legacy systems are mapped, the job becomes less onerous. There are superb facilities for enabling schemas to be compared. They make it very simple to see what is being changed by new versions and to incorporate them in the CDM. Whatever is spent on the task is recovered more than ten-fold in the reduction in the cost of data archaeology. The Open Data Model® CDM lowers the risks associated with making changes. Unexpected consequences resulting from poor information are no longer the occupational hazard they once were.

The reason that the maintenance burden begins to lift is that people that are responsible for making change are now well informed. These people will use the information in the Open Data Model® to plan changes. The fact they have complete and accurate information at their fingertips will significantly improve their productivity. These users of the metadata will not need to understand the mapping process; they will deal with the information through browsers that show them the results of the effort that went into creating and populating the repository.

Appendix

Figure 12 shows the overall flow of the modeling and mapping activities:

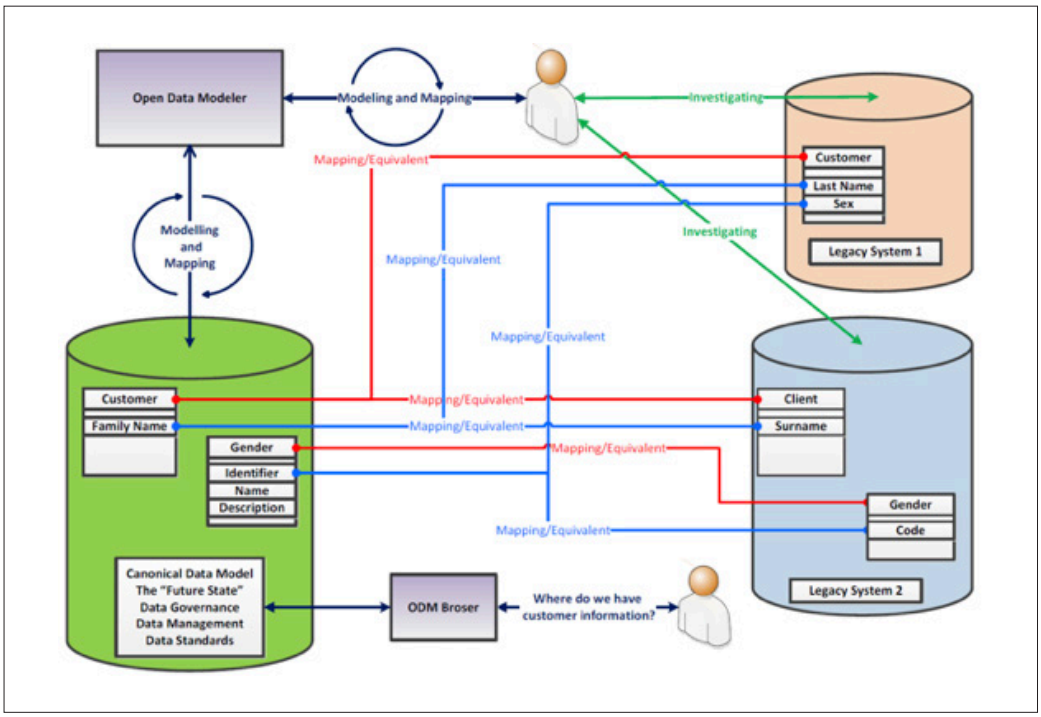


FIGURE 12 – THE ODM FLOW