Metadata Management for Data Warehouse Projects

Stefano Cazzella
Datamat S.p.A.
stefano.cazzella@datamat.it

Abstract

Metadata management has been identified as one of the major critical success factor in data warehousing projects, but the industrialization process of metadata management did not yet reach a satisfying level of maturity from both a methodological and a technological point of view; therefore every project has to build its own metadata management system following the methodology that better fits its requirements.

This paper describes the metadata management methodology designed during a e-government data warehouse project and shows how it could be applied by means of a real case study. The methodology we have devised and effectively put into action covers the entire process of building and running a metadata management system in a data warehouse project; its main objective is to exploit metadata to create a homogeneous, structured and integrated “vision” of the data warehouse system in order to produce valuable metadata-based services for the end users.

1. Introduction

Metadata is a wide and very pervasive subject matter that is rapidly gaining interest in the last years. In this paper we restrict our focus to metadata management in data warehousing contexts; anyway some of the considerations and the approaches proposed in this paper could be generalized to other application fields.

Metadata has been recognized as a relevant topic since the beginning of data warehousing and it is still considered as one of the major critical success factor in data warehousing projects [1][2][3]. Indeed metadata management is crucial to improve overall project quality and flexibility, as to increase the usability of the system by the end users.

Despite the attention addressed by all the operators of the data warehouse industry (technology vendors, consultant companies, etc.), the industrialization process of metadata management did not yet reach a satisfying level of maturity neither from a methodological nor from a technological point of view [3]. Today, we still does not have a ready to go metadata management solution that can be applied in every data warehouse project “as-is”. Conversely, almost every project has to build its own metadata management system from scratch or integrating different commercial tools with custom piece of software [4][5]. These “customized systems” are often realized in order to accomplish specific and ambitious requirements generated by the increasing expectations related to metadata management. Usually these requirements grow in time (iteration by iteration) as you achieve the first results in exploiting metadata.

The main goal that should be pursued by a metadata manager is to create a structured, integrated and homogeneous vision of the whole data warehousing system using the deliverables produced during the life cycle of every project iteration. This goal is the foundation that makes a lot of metadata requirements reachable. If you have a structured, integrated and homogeneous vision of the whole system you can analyze the system itself and provide useful information about the system to all the actors involved in the data warehouse project: from the data warehouse administrator, to the end users of the system.

More and more often the data warehouse architecture includes a “Information Portal” that represents the web interface used to deliver information and analytical services (e.g. OLAP, statistical analysis, KDD, etc.) to the end users of the data warehousing system. This portal is an ideal companion for the metadata management system; exploiting its web based technologies, it simplifies the integration between data and metadata realizing the well-known adage: data + metadata = information. Indeed metadata could be effectively exploited to help users to easily access, use and understand the business data provided by the data warehousing system [6]; support functionalities like these increase dramatically the success probability of the project, since it is closely correlated to the real value the users assign to the system.

Many commercial tools and methodologies try to cope with metadata management from a technological perspective: they focus their attention on the metadata repository and its logical structure. The same issue
could be addressed in a more effective and easier way moving from a logical to a conceptual representation of metadata. Reasoning at a conceptual abstraction level gives you the flexibility you need to manage metadata in a “double dereferenced way” working dynamically both on metamodel and metadata. This approach has been largely used in the formulation of the methodology and in the development of the metadata management system shown in this paper.

2. The methodology

The metadata management methodology we have devised identifies the main goals and the macro activities that should be part of the metadata management process of a data warehouse project. The methodology is independent from any technological issue and then could be implemented using a COTS approach, an home made one or a mix of them. It is independent from the architecture of the metadata management system, too.

Our methodology is structured in four phases:

1. Metadata sources identification. In this phase all the potential metadata sources, that are involved in the developing process of the data warehouse system, are identified. Every source is analyzed to define its metamodel.

2. Metamodel integration. In this phase we integrate the metamodels defined in the previous phase to get a unique model that is able to represents the whole system. The integrated metamodel is the composition of the metamodels of the selected metadata sources.

3. Metadata integration. In this phase the metadata produced by every source are integrated to recompose an homogeneous view of the data warehouse system accordingly to the integrated metamodel built up in the second phase.

4. Metadata analysis and publication. In this phase the integrated metadata are used to carry out analysis of the data warehouse system (i.e. cross-impact analysis, data lineage, etc.) for the technical users and to create advanced functionalities that help the end users to better exploit the system and the information it provides.

Usually a data warehouse system is developed iteratively to reduce risks and deal with the complexity of a data warehouse project [7]; this methodology offers the same flexibility.

The four phases of the methodology could be applied iteratively to direct a gradual broadening of the metamodel and, consequently, the metadata handled by the system. This could be useful whether you plan to smoothly add metadata management to an already running data warehouse system, or whether you introduce new metadata sources (e.g. you change your production process adopting a new CASE tool or establishing a new deliverable) during the iterative building of a data warehouse system.

It’s also possible to reapply iteratively only the last two phases in order to update the metadata on every iteration of the life cycle of the data warehouse system; in this way the metamodel is still the same, while the metadata will change accordingly to the last version of the system.

2.1. Metadata sources identification

The goal of this phase is to identify all the metadata sources that will be handled by the metadata management system and specify their metamodels. The metadata sources should be chosen among the different deliverables produced during the life cycle of the developing process, form the analysis phase to the test. Valid examples of metadata sources are: the conceptual data model, the logical data model, ETL procedures, business rules, user specifications, etc.

The more tractable sources are the deliverables produced by CASE and developing tools, since they already have their own metamodels. To extend the metadata managed by the system to the information contained in other deliverables, new metamodels must be defined to specify the logic structure and the semantic of every piece of information the deliverables contain.

At the end of this phase all the sources must have their own metamodel. Every metamodel should be represented using a single modeling language; since the proposition of CWM [8], metamodels are commonly represented using UML class diagrams. In the following we assume that every metamodel is represented by a set of classes, and a set of associations between classes; the two sets define a graph where the classes are the vertexes and the associations are the edges. Every vertex is labeled with a full qualified name composed by the metamodel name and the class name, and has a set of attributes that are the attributes of the class. Every edge is adorned with the roles of the participating classes in the corresponding association and, eventually, a name; moreover it should be classified in “Depends-on association”, “Part-of association”, etc. to enable impact analysis, data lineage, and other kind of analysis. Using UML this classification could be represented by stereotypes.

During an iteration of a data warehouse project, these metamodels are progressively instantiated as the deliverables are produced and released; these instances are the metadata that will be handled by the metadata management system.
2.2. Metamodel integration

The goal of this phase is to integrate the different metamodels identified in the previous phase. The integrated metamodel has to represent an homogeneous view of a data warehousing system by the merger of the metamodels of the deliverables that contribute to build the data warehouse system. The integration process is performed by the introduction of new associations between the classes of different metamodels; so every source metamodel maintains its identity and integrity. No metamodel should remain isolated; the classes of the integrated metamodel must build up a connected graph. This is the integrated metamodel graph.

In order to find out which association could be added between the classes of the metamodels, it’s better to follow a two step strategy. First you have to consider the set of source metamodels only; from this set you have to couple all the interrelated metamodels according to the architecture of the data warehousing system and the methodologies applied in the analysis, design and development phases. You get a graph whose vertex set is the set of metamodels and where every interrelated couple of metamodels is an edge; this graph should be a connected graph. Since every deliverable produced during the life cycle of the system takes part to the building of the system, if you have included enough deliverable metamodels during the first phase of the methodology, you should be able to get a connected graph. Then, as the second step, you have to look into every couple of interrelated metamodels to identify the associations between the classes of the two metamodels that represents the relationship of the couple. If every source metamodel is connected, the integrated metamodel graph will be connected, too.

One of the more common kind of association used to integrate metamodels is an equivalence relation; this kind of association is used when two classes of two different metamodels represent the same type of object. For example, a table of a relational data base is represented by a Table class in both the data modeling metamodel and the ETL metamodel; these two metamodels could be integrated introducing an association between the two Table classes that asserts their equivalence (but not their equality).

2.3. Metadata integration

The goal of this phase is to integrate the metadata of a specific project (or an iteration of a project) to build up an homogeneous view of the particular data warehouse system realized by the project itself. The integration is performed instantiating the integrated metamodel defined in the previous phase. This process could be completed incrementally: every source metamodel is instantiated as the corresponding deliverable is consolidated; the associations between two different metamodels are instantiated when the two metamodels are instantiated. In this way the metadata could be regularly integrated during the life cycle of the project in order to support the following activities of the life cycle. Most association instances that integrates metadata of different metamodels could be derived applying some integration rules. For example, the equivalence association between two classes could be automatically instantiated when two instances of these classes has the same full qualified name; this rule could be applied to the instances of Table classes of the data modelling and ETL metamodels when the instances represents the same relational table, that is when they have the same name and belongs to the same data base (or schema). Usually not all the association instances needed to integrate the metadata could be derived by integration rules; in these cases you have to provide all the information needed to integrate the metadata by yourself. There are many ways to do this and they depend on the technical architecture of the metadata management system.

At the end of this phase you get a new graph, the metadata graph, whose vertexes are the instances of the classes of the integrated metamodel and the edges are the instances of its associations (both the internal and the integrating ones). Indeed, this graph is an instance of the integrated metamodel graph defined in the previous phase and, if the metadata have been correctly integrated, it is a connected graph.

From a technical/technological point of view, the integration could be carried out in different ways; in effect, no one is excluded a priori. You could choose a centralized architecture, with a unique metadata repository where you store the metadata extracted from different sources (the deliverables); the source metadata could be accessed directly or by means of interfaces provided by the tools used to create the deliverables. Otherwise you could decide for a distributed architecture where all the repositories of the different tools involved in the production process are federated; since the associations between different metamodels are not included in any of the data model of the source repositories, the metadata are merged using an additional repository to store only the instances of these associations. Anyway the most important feature of the architecture you choose is the availability of an interface that enables you to access all the metadata in the same way; you should be able to navigate among the metadata graph knowing only the integrated metamodel and the common interface. Often this interface is based on a sort of meta-meta-model that is used to represents all the metamodel instances handled by the metadata management system.
Moreover you could choose to build up the metadata management system from scratch, to assemble it using commercial components or to buy a commercial tool. In the last case, be sure the chosen metadata management system gives you the possibility to redesign your own metamodel. Commercial components and tools are generally useful to get out metadata from the source tools and to conduct the classic cross-impact analysis, but often system integration activities are required to fulfil all the requirements a metadata management system usually have.

2.4. Metadata analysis and publication

This is a crucial step in the methodology, since all the effort spent to handle and integrate the metadata during the first three steps of the methodology is justified by the return earned by the analysis and the applications produced in this phase. The metadata graph enables to perform the classical cross-impact analysis and data lineage analysis; moreover the same graph could be used to evaluate software quality metrics; e.g. “how many relational tables have not been derived from an entity of the conceptual data model?” or “what is the average number of mappings per target table?”. These kind of analysis are addressed to technical users as workgroup members or data warehouse administrators; there is at least another kind of user that may be supported in its daily activities using metadata: the end user. Indeed there are a lot of interesting services that may be provided to the end users exploiting the information collected in the metadata graph. First of all the metadata graph itself could be rendered to enable end users to move around metadata. For example, every node of the graph could be rendered as an HTML page and every arc is implemented as an hypertext link between two pages. In this way every user could browse the graph as a web site. This pool of pages represents a sort of encyclopaedia about the system; with a good research engine you could index this pages to give the end users the possibility to perform text based searches. But, the most effective search capabilities may be realized building simple “wizard” applications that drives the end users in their search. These kind of applications starts the search from the most general context of analysis and reduce step by step the search space steering to the search target. Finally, the metadata graph could be used to produce technical documentation and manuals with a very low effort. For example you can get a list of the reports provided by the system with their descriptions and the sets of measures and dimension they contains.

The metadata-based services (wizard and simple search engine), the metadata graph rendered as HTML pages, the manual and so on could be published on the Data Warehouse Portal together with the reports, the OLAP environments and the other functionalities provided by the system. The integration between data and metadata could be further carry out; e.g. every report should be coupled with the corresponding metadata in order to associate to the report a meaningful description of the data it contains and ensure that the users has all the possibilities to correctly understand the meaning of the numbers shown by a report. The publication of metadata and metadata-based services together with classical data and data-based services (e.g. OLAP, statistical analysis, etc.) enhances the usability of the system from the end users perspective and this is a crucial aspects that may significantly increase the success probabilities of a data warehouse system on the long run.

3. The case study

The case study describes our successful experience in metadata management accrued during a complex data warehouse project for e government. During this project we have devised the methodology we discussed in the previous paragraphs, we have built up a metadata management system that supports this methodology and we have iteratively applied this methodology accordingly to the evolution of the whole data warehouse system.

3.1. The sources’ metamodels

The data warehouse has the classical data flow shown in the left side of figure 1. To rebuild an integrated vision of a data warehouse, every component should be considered, especially the ones involved in the data flow architecture. So we have considered which deliverables were released for every component during the development life cycle (see right side of figure 1):

- “Databases” component - the logical/physical data models of the staging area, of the enterprise data warehouse (EDW) and of the data marts (DM);
- “ETL” component - the procedures used to load the data warehouse (EDW ETL) and the data marts (DM ETL);
- “Front-end” components - a semantic layer, that provides a business representations of the data stored in the data mart physical data model, and a pool of predefined reports.
All these deliverables were selected as metadata sources. To complete the picture we also included among the metadata sources the logical/physical data models of the source operational systems.

The deliverables we listed above were released by the activities of the Design and Development phase of the development life cycle. As shown in figure 2, during the Definition phase two very valuable deliverables are produced: the conceptual data model, that represents all the information handled by the system independently from technical issues, and the dimensional fact model (DFM) [9], that was used to represent the end user’s requirements in terms of multi-dimensional analysis capabilities. These models represents the whole system from a business point of

**Figure 1. Metadata sources**

**Figure 2. Development life cycle**
view: they represents the available information with their business meaning and the user’s analysis capabilities. These deliverables, together with the description of the derivation process of logical data models from the conceptual one (C2L), completes the metadata sources we considered for metadata management.

3.2. The integrated metamodel

Every deliverable has its own metamodel; it could be the metamodel of the CASE tool used to realize the deliverable or a metamodel specifically designed to represent it in a structured form (see table 1).

![Integrated metamodel](image)

Figure 3. Integrated metamodel

These metamodels have been integrated following the indications provided in the “Metamodel integration” paragraph. The integrated metamodel has been designed in order to represent, in a structured form, the whole data warehouse system, as illustrated in figure 3. The arcs between the nodes of this connected graph represents the associations used to integrate the different metamodels. For every couple of metamodels linked by at least one arc, one or more associations between classes of these metamodels have been defined in order to integrate the sources’ metamodels. At the end of this process we got an integrated metamodel that may represent every data warehouse that shares the same development methodology and technical architecture (i.e. different iterations of the same data warehouse).

3.3. The metadata management system

The metadata management system has been realized starting from a commercial metadata management tool. Its functionalities has been extended and integrated with other custom applications in order to realize a system that was able to cope with the client’s requirements.

The first kind of software applications we had to realize are the ones needed to represent the metadata of the deliverables produced without the support of a commercial CASE tool (see table 1). These applications help analysts and developers to document their work in a common, standardized way also when there aren’t a specific tool for that work; so important notions about the system, that otherwise would be retained only in the mind of one person, could be shared among different people and used as metadata.

The other software application is the core of the metadata management infrastructure we realized; it is in charge to represent the integrated metadata using an homogeneous meta-meta-model and produce the metadata based services that have been published on the Information Portal.

The acquisition and integration of all the metadata produced by commercial tools and the metadata analysis was performed using the commercial metadata management tool.

Table 1. Metamodels

<table>
<thead>
<tr>
<th>Metamodel</th>
<th>Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conceptual and logical/physical data model</td>
<td>Commercial CASE tool with its own metamodel; it exports its metadata in XML (with proprietary DTD).</td>
</tr>
<tr>
<td>C2L</td>
<td>Custom application; the metamodel is a subset of a classical E/R metamodel</td>
</tr>
<tr>
<td>Semantic layer</td>
<td>Commercial OLAP system with its own metamodel; it exports its metadata using public non-standardized APIs.</td>
</tr>
<tr>
<td>DFM</td>
<td>Non-commercial CASE tool with its own metamodel; it exports its metadata in XML.</td>
</tr>
<tr>
<td>ETL procedures</td>
<td>Commercial ETL tool with its own metamodel; it exports its metadata by means of public non-standard relational views provided to access its private repository.</td>
</tr>
<tr>
<td>Reports</td>
<td>Custom application; the metamodel has been designed ad-hoc to represents both report’s content and its structure.</td>
</tr>
</tbody>
</table>
3.4. The metadata-based services

The main metadata-based service that is published on the data warehouse information portal is the rendering of the integrated metadata graph as HTML pages. It is a large connected graph with more than 100,000 nodes. A little subset of this nodes are proposed as entry point for browsing metadata; some examples of entry point nodes are the nodes representing reports, the conceptual data model, the dimensional fact models, and so on. Starting from any of these nodes, every other node is reachable; so, for example, you can browse from an entity of the conceptual model to the reports that contains the data modeled by this entity.

Since browsing the metamodel may be a bit too difficult for some kind of non technical users, another category of metadata based service has been realized. They are search wizards that helps the users to find the information they need during their daily working activities. These wizards start from the context analysis of a semantic layer and give the user the possibility to choose the one that interests him. Then they propose all the measures that belong to that context; and at the last step the wizards show all the reports that contains the selected measure. Then the user can open directly the report he was looking for. These wizards are very useful for new users that still don’t know all the information provided by the data warehousing system; moreover they are useful when new information (or a new system) are provided to help the users to familiarize with them.

5. Conclusions

The distance between the requirements about the exploitation of metadata to help the end users to get the best from the data warehousing system and the solutions proposed by metadata industry led us to devise a new methodology for metadata management. Many methodologies and tools are focused on collecting and integrating all the metadata in a common repository or in a federated one; in our methodology we define the metadata integration strategy at an higher level of abstraction working with a shared metamodel. The main goal is to exploit the deliverables released during the project life cycle in order to create an homogeneous, structured and integrated “vision” of the data warehouse system and produce valuable metadata based services for the end users. This methodology has driven the implementation of a flexible metadata management system, built up combining commercial products and custom applications.

The data warehouse system, including the metadata management component, has been recently subjected to an external audit; the auditing results assert that one of the aspects that sticks out as being especially well done is the metadata surrounding the data warehouse and that “the metadata infrastructure that has been created is truly remarkable”.

This paper has shown how we realized a technical infrastructure for metadata management founded on a sound methodology, but we didn’t talk about organizational and cultural issues involved in metadata management. These aspects are as important as the technical ones in order to realize a successful metadata management system, but they are more insidious to cope with. Metadata management may require some extra effort, e.g. to document exhaustively every attribute of every entity, every ETL mapping, every cell of every report and so on, but the returns in terms of quality, maintainability and usability of the data warehousing system largely cover any extra cost related to metadata management activities.

5. References