Information Integration in a Global Enterprise Some Experiences from a Financial Services Company

Robert Marti

Swiss Re
Mythenquai 50/60
CH-8022 Zurich, Switzerland
robert_marti@swissre.com

Abstract: In most commercial enterprises, information is scattered across a large number of (legacy) data stores. Moreover, it is nearly impossible to obtain funding to replace these data stores with a single integrated database, given tight budgets, time-to-market pressures and past failures of various kinds of large-scale efforts. In this paper, we advocate an approach which hinges on (1) a high-level enterprise information architecture, (2) the identification of the stable elements in this information architecture – essentially identifiers of some core entity types, the sets of legal values of some descriptive attributes – and (3) a central metadata and reference data repository which supports the tracking of the history of reference data.

1 The Application Area: Reinsurance

A reinsurance company, or *reinsurer*, insures insurance companies, also called *(primary) insurers*. Much like insurers assume risk for a fixed price from their clients, either persons or organizations, reinsurers assume risk from primary insurers. Depending on the their needs, primary insurers mainly buy reinsurance to either assume a larger portfolio of similar small risks, and/or to pass on parts of a specific very large risk. For further information on the subject of reinsurance, we refer the reader to [CL00].

While both insurers and reinsurers assume risk, they are different in that a primary insurer is a retailer with a relatively large customer base and relatively few highly standardized products, while a reinsurer is a wholesaler with a relatively small customer base and relatively large number of products specifically designed for one client. As a result, the volume of data and transactions is not nearly as much of a challenge for a reinsurer as it is for a primary insurer, or a retail bank, for that matter. On the other hand, given product diversity, the size (and therefore clout) of many clients, and the global reach of the business, standardization is much more difficult on the data side as well.

2 The Challenge: Legacy and Cost/Time Constraints

In the area of information management and databases, two major differences between working in almost any commercial enterprise as opposed to academia are:

- *Historical baggage*
  Most large, global enterprises look back over a history of legacy systems and associated data stores, often developed in isolation, without adequate documentation, using yesterday's methodologies and technology. Many of these applications still form a vital part of the operational backbone of the business and are as such highly valuable. In other words, they represent a huge investment which cannot easily be replaced.
- *Pressures with respect to cost/benefit and time-to-market*
  In today's corporate environment, obtaining funding for large, long-term projects is increasingly difficult due to cost and time-to-market pressures. Often, the best chance is to find an in-house client with budget and then catering to his specific needs, which in turn may conflict with the interests of the corporation as a whole.

3 Application Landscapes

In order to identify both gaps and (potential) overlaps in applications fielded in many different locations and supporting various reinsurance products, Swiss Re has been using a simple matrix called an *application landscape* for over 5 years:

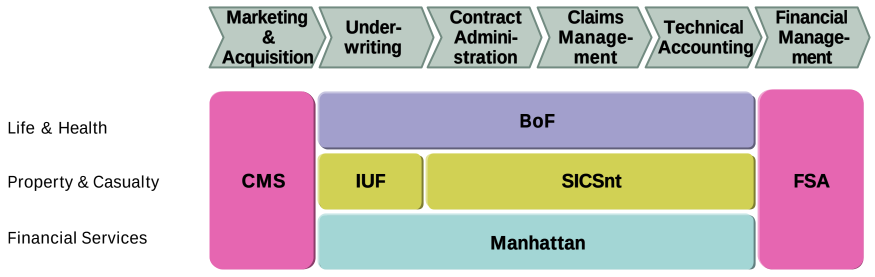

Figure 1: (Simplified) Application Landscape

The column headings represent a high-level value chain which is roughly structured into a sequence of core business processes (e.g., “Marketing & Acquisition”, “Underwriting”), while the rows are typically associated with either locations, organizational units, products, or a combination thereof (e.g., “Life & Health”, “Property & Casualty”). If an application supports one or more business processes by capturing the data which is produced during the execution of those processes, then the application is depicted underneath those processes. For example, the application CMS (Client Management System) supports data capture in the “Marketing & Acquisition” process across the entire group, while SICSnt supports data capture of the processes “Contract Administration”, “Claims Management”, and “Technical Accounting”.

Note that drawing an application landscape is not an exact science in that

- the business processes of the value chain do not necessarily occur in the sequence shown
- the boundaries of what an application supports exactly is sometimes a little bit “fuzzy”.

In the context of the Framework for Information Systems Architecture pioneered by John Zachman [Za87, SH92, Co96, Ha03][1], application landscapes are on the level of the business owner and essentially relate activities with locations (represented by applications).

4 Architectural Framework and Piecemeal Development

Application landscapes have proven to be a useful tool for communication to the business side and top management, as well as to align development plans of different organizational units. Moreover, application landscapes highlight the need for application integration:

While the need for integrating the various information islands both along the value chain (see Figure 2) as well as across product lines and/or organizational units (see Figure 3) is unquestioned, it is nearly impossible to make a sound business case for “big bang” approaches such as developing (detailed) Enterprise Data Models or even a single integrated Enterprise Data Warehouse.

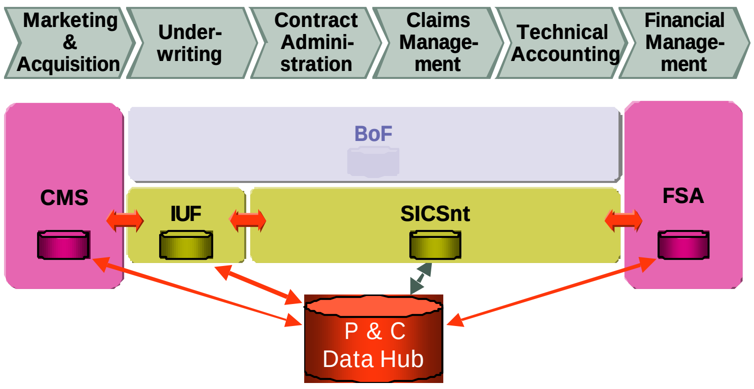

Figure 2: Integration along the Value Chain

[1] Following [Ha03], the Framework for Information Systems Architecture is a matrix with 5 layers addressing the different views of various stakeholders (planner, business owner, architect, designer, builder) and 6 columns addressing the different aspects of an information system (data, activities, locations, people, time, motivation).

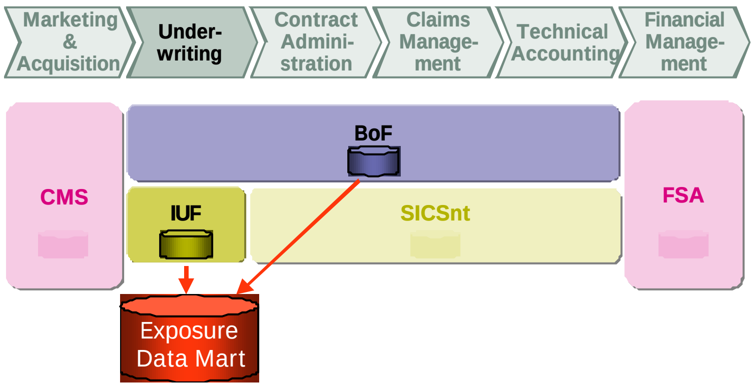

Figure 3: Integration across Organizational Units / Products / Location

A compromise is to adopt an approach which hinges on the following two elements:

- *A High-Level Architectural Framework*
  Instead of developing complete and detailed data and business process models in multi-year projects, enterprise-wide models are only elaborated as far as needed in order to get a rough overall picture, stopping at approximately a dozen subject areas (high-level entity types) and a dozen core business processes. In the context of the Zachman Framework, data and business process models are elaborated to serve the business owner's and the business architect's needs, but not down to a (system) designer's view.
- *Piecemeal development and integration*
  New solutions are developed within the confines of the high-level blueprint, based on a sound business case, in a piecemeal fashion. Details are added to the high-level models on a "just-in-time" basis, which in rare cases may even lead to amendments in existing parts of the models.

5 High Level Data Model

In order to establish a high-level architecture which allows subsequent information integration with a minimum of disruption to the existing systems and databases, the first issue is to identify the stable elements of the overall data model.

Swiss Re's high-level data model was developed in a two-pronged approach:

- Published generic and industry-specific data models, see e.g. [Ha96, Fo97, Si01], and architectures, IBM's Insurance Application Architecture (IAA, see [IB02]) and a similar application architecture developed within the German insurance industry (VAA, see [GD96]) were studied.
- At the same time, the data models of existing systems as well as the contents of various reports were analyzed in a bottom up way.

A simplified version of the basic data model, representing the business owner's view in Zachman's framework [Ha03], was established in late 1997 and looks as follows:

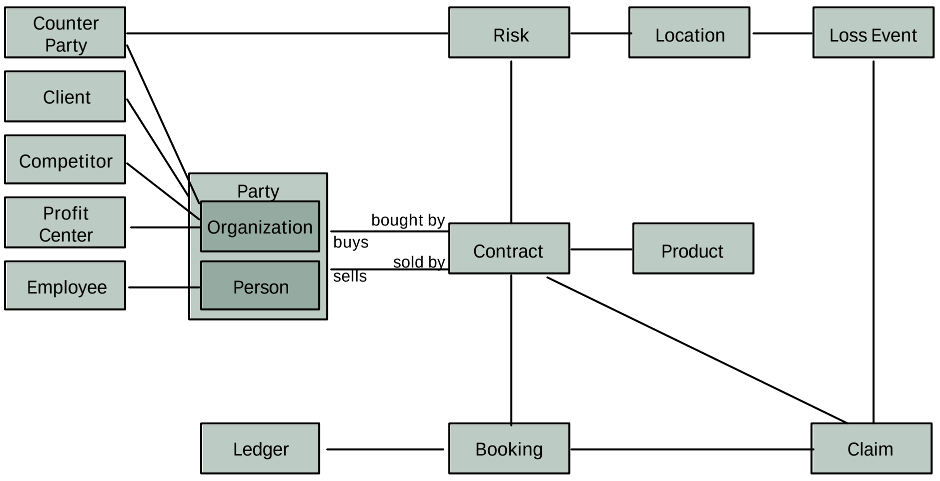

Figure 4: High-Level Data Model for Reinsurance

Note that relationships in this diagram are generally considered to be many-to-many relationships, with the exception of the relationships between entity type Party and its subtypes on one hand and the entity types which establish the roles that can be played by a Party on the other. This data model has served as a foundation to design the physical data model of both a Data Warehouse project as well as a new project to establish an Operational Data Store (ODS) which mainly serves as a data exchange hub.

6 Stable Elements of a Data Model

The entity types in a data model can be classified according to their usage as follows (see also Figure 5):

- *Interaction entity types* are entity types which describe business interactions between the enterprise and an outside entity such as a client, a supplier or some sort of intermediary. Examples of interaction entity types are Quote, Contract, Booking. (Note that interaction entity types contain information which is typically stored in the fact tables of a data mart [KR02]. Most importantly, they comprise key business performance indicators, e.g., in reinsurance, premium_earned, losses_incurred, and cost.)
- *Context entity types* are entity types which describe the context of business interactions. Examples of context entity types are Product, Client, Employee, ProfitCenter. (Context entity types contain data which is typically stored in dimension tables of a data mart [KR02].)

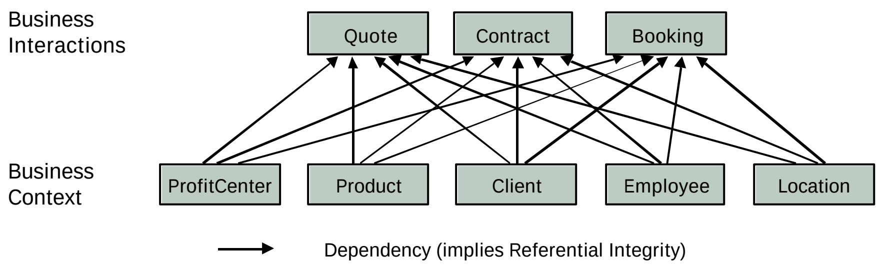

Figure 5: Interaction Entity Types depend on Context Entity Types

The stable elements of an enterprise information architecture are to a large extent determined by the structure and (predefined) content of context entity types:

- *Identifiers of context entity types*
  Global identification schemes of context entity types serve as the glue to tie together related information scattered across data stores controlled by the various applications. Examples are enterprise-wide IDs for clients, employees etc. These identifiers serve as surrogates for real-world entities and as such (1) must be devoid of any semantics, (2) must not change during the lifetime of the corresponding real-world entity, and (3) must not be reused for another entity.
  Note, however, that establishing a global identification scheme alone is not good enough. For example, even with a single globally used client management system such as Swiss Re's CMS, it is still possible, depending on corporate culture and established processes, that one client is represented multiple times in the database.

- *Domains of attributes of context entity types*
  In order to segment the values of key business performance indicators according to the properties of context entity types, a common terminology has to be established. An important part of this terminology is defined by the *domains* of attributes which describe context entity types (see also Figure 6). Typically, these domains look like enumeration types in programming languages such as C or Pascal in that they consist of a predefined set of data values.
  In addition, within a domain, there are often taxonomic relationships between data values in that a data value representing a more general concept is related to a number of data values representing more specific concepts. While data is usually entered into OLTP systems using the most specific concepts of a domain only, terms describing more general concepts are typically used for analysis and reporting purposes.
  Example (see also Figure 7): The domain LineOfBusiness consists of a predefined set of terms (enumeration constants) such as 'property', 'casualty', 'engineering', 'marine', 'life', 'health' etc. However, smaller lines such as 'engineering' and 'marine' may be subsumed by the more general term 'special line'.

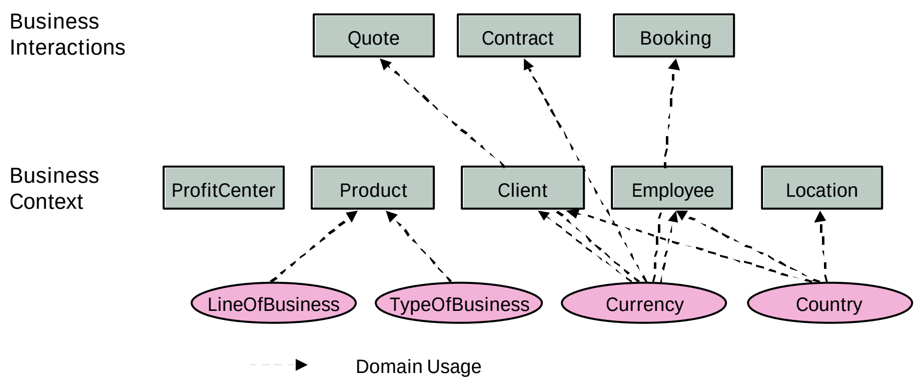

Figure 6: Domains and their Usage by Interaction and Context Entity Types

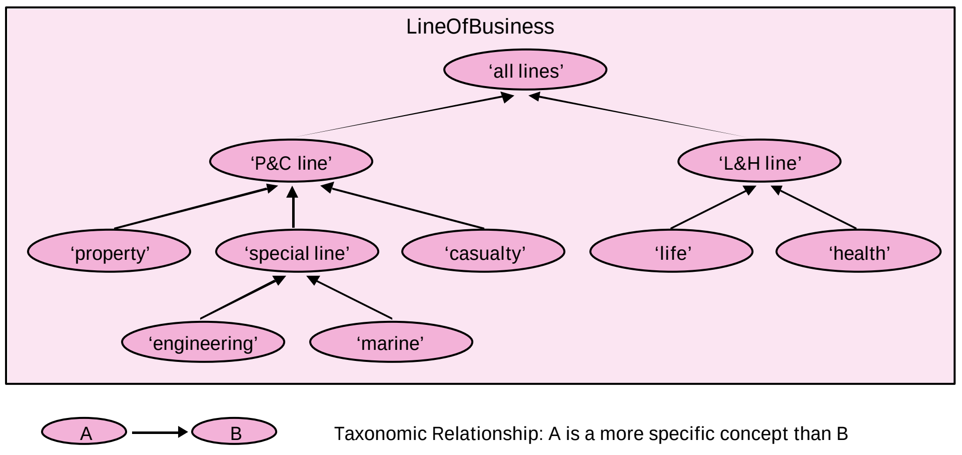

Figure 7: Taxonomic Relationships between Data Values of a Domain

7 Repository for Metadata and Reference Data

Regarding the integration of enterprise applications, there have been many proposals for technologies to solve interoperability issues, e.g., CORBA, J2EE, Web Services, etc. While these technologies are great enablers, they cannot directly solve the much more important and difficult issues to achieve interoperability on a semantic level. (This is even true of XML, which is being touted as the solution to all interoperability problems, beacuse the real issue is to agree on the semantics of the XML tags used to describe information which is being exchanged.)

Since migrating existing operational applications from current local identification schemes and terms to their global counterparts is often non-trivial, mappings from local identifiers and terms to the corresponding global identifiers and terms must be supported.

Moreover, a *central repository for metadata and reference data* should contain the definitions of terms used to describe data structures and (predefined) content, their usage in applications, and mappings between these terms.

As an aside, note that it is sometimes difficult to clearly distinguish metadata and reference data: Typically, metadata is defined as "data that describes or specifies other data". As an example, the names of tables and columns in a relational database setting are considered metadata. Reference data is defined in [Ch01] as "any kind of data that is used solely to categorize other data found in a database, or solely for relating data in a database to information beyond the boundaries of the enterprise", i.e., consists of domains and predefined data values as discussed in the previous section. However, depending on the design of a database, what is metadata in one design can be reference data in the other, as illustrated by the terms on gray background in the following figure which depicts two alternative database designs:

GrossNetV1

Treaty Year	Business Year	LoB	ToB	Premiums	Claims
1999	1999	'Property'	'proportional'	1654	-1623
...	...	...	...	...	...

GrossNetV2

Treaty Year	Business Year	LoB	ToB	Key Figure Name	Key Figure Value
1999	1999	'Property'	'proportional'	'Premiums'	1654
1999	1999	'Property'	'proportional'	'Claims'	-1623
...	...	...	...	...	...

Figure 8: The same business terms may appear as metadata or as reference data

The ultimate goal is to administer all metadata and reference data used in various systems in the central repository so that it serves as a single point of reference for all applications. This central repository should also keep track of the *history of reference data* in order to support the analysis of key business performance indicators over time, notwithstanding changes to the attributes of context entities, e.g., as induced by the change of location of a client, or an organizational realignment.

The current tool for the management of reference data at Swiss Re, SDL Tool Release 3, only supports (1) the definition of business terms, (2) browsing and querying available business terms over the intranet, and (3) the generation of a manual which specifies the structure for the collection of financial data. With one exception, reference data is currently still transferred to operational systems in a manual process. Also, there is no support for historization yet. The system is built on top of relational technology (DB2 on OS/390), given that with a few exceptions, almost all key applications within Swiss Re are also built on relational technology (DB2 on OS/390 or Oracle on Solaris).

The design and implementation of a new version of SDL Tool (R4) has been started. It will contain various improvements on a technical level, most importantly regarding the release process of consistent sets of reference data, including various APIs to allow other applications to retrieve reference data from the central repository, and support for versioning and historization of metadata. An excerpt of the SDL R4 data model is shown in Figure 9.

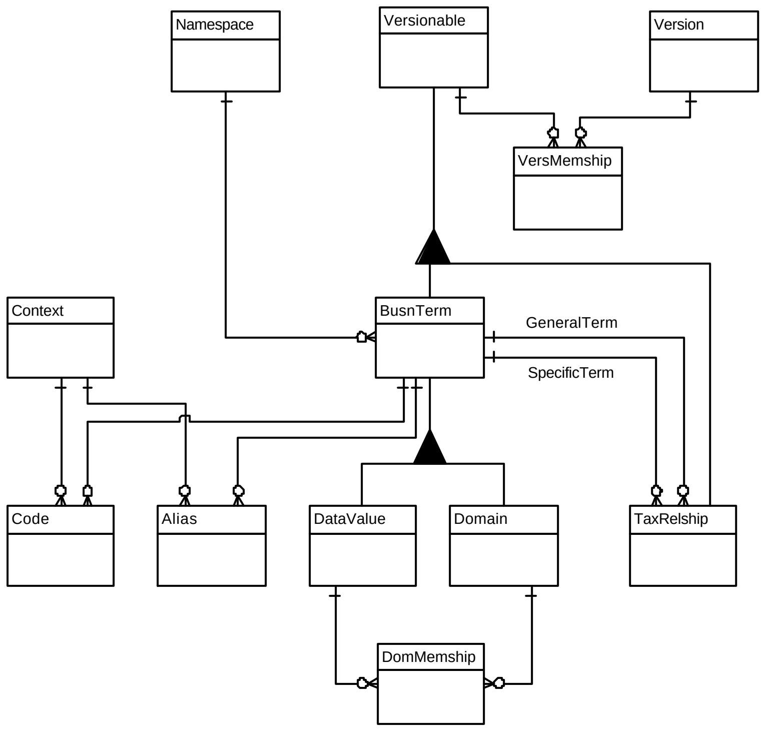

Figure 9: Schema of SDL Tool R4 (Excerpt)

With respect to the historization of reference data, we'd like to mention in passing that having a temporal query languages such as TSQL2 or SQL/Temporal [Sn00] is not nearly as important as establishing a good design of a data model which supports historization. The basic idea is to represent the lifespan of an entity together with its "unchanging" attributes in one table, and each group of attributes which changes simultaneously in a separate table (see e.g. [Ma94], or, more accessibly, [Sn00] Chapter 11 and [DD03] Chapter 10 for a discussion of design of temporal databases).

8 Conclusions

In summary, information integration in a global enterprise is based on (1) a high-level data model, (2) global identification schemes of relevant (context) entity types, and (3) a global terminology for the relevant domains of important attributes, where relevance is to a large extent established by global business needs. Of course, even such a minimalist approach to information integration is not without challenges. As [MK01] eloquently point out, success of IT projects and systems in corporations depends not only on having the right technology, but also on having the right content, and, most importantly, an information culture which encourages information sharing across organizational boundaries.

Acknowledgements

The author would like to thank Benno Büeler and Hans Wegener for their work on SDL Tool R4, and P. Cartwright, T. Harris, M. Kaufmann, A. Mönkeberg for their contributions to the topic of metadata and reference data management in general and earlier releases of SDL Tool in particular.

References

[Ch01] M. Chisholm: *Managing Reference Data in Enterprise Databases*. Morgan Kaufmann, 2001.

[CL00] R. L. Carter, L. D. Lucas, N. Ralph: *Reinsurance, 4th Edition*. Reactions Publishing Group / Guy Carpenter & Co., 2000.

[Co96] M. A. Cook: *Building Enterprise Information Architectures: Reengineering Information Systems*. Prentice Hall, 1996.

[DD03] C. J. Date, H. Darwen, N. A. Lorentzos: *Temporal Data and the Relational Model*. Morgan Kaufmann, 2003.

[Fo97] M. Fowler: *Analysis Patterns – Reusable Object Models*. Addison Wesley, 1997.

[GV96] GDV (Gesamtverband der Deutschen Versicherungswirtschaft e.V.): *VAA – Die Anwendungsarchitektur der Versicherungswirtschaft*. VDS Dienstleistungs-GmbH, 1996.

[IB02] IBM Corp.: *IBM Insurance Application Architecture (IAA)*. http://www-1.ibm.com/industries/financialservices/doc/content/solution/278918103.html

[Ha96] D.C. Hay: *Data Model Patterns – Conventions of Thought*. Dorset House, 1996.

[Ha03] D. C. Hay: *Requirements Analysis: From Business Views to Architecture*. Prentice Hall, 2003.

[KR02] R. Kimball, M. Ross: *The Data Warehouse Toolkit, 2nd Edition.* John Wiley & Sons, 2002.

[Ma94] R. Marti: Entwurf Temporaler Datenbanken. *Fortbildungskurs Temporale Datenbanken*, ETH Zürich, 1994.

[MK01] D. A. Marchand, W. J. Kettinger, J. D. Rollins: *Making the Invisible Visible: How Companies Win with the Right Information, People and IT.* John Wiley & Sons, 2001.

[Si01] L. Silverston: *The Data Model Resource Book* (Vols. 1 and 2). John Wiley & Sons, 2001.

[SH92] S. H. Spewak, S. C. Hill: *Enterprise Architecture Planning: Developing a Blueprint for Data, Applications and Technology*. John Wiley & Sons, 1992.

[Sn00] R. T. Snodgrass: *Developing Time-Oriented Database Applications in SQL.* Morgan Kaufmann, 2000.

[Za87] J. A. Zachman: A Framework for Information Systems Architecture. *IBM Systems Journal*, Vol. 26, No. 3, 1987.