Editor: Siobhán Clarke • siobhan.clarke@cs.tcd.ie

Model-Based Data Engineering for Web Services

Andreas Tolk and Saikou Y. Diallo • *Virginia Modeling Analysis & Simulation Center, Old Dominion University*

Although XML offers heterogeneous IT systems a new level of interoperability, it doesn't ensure that the various systems correctly interpret the data they receive. To address this, data engineering supports clear definitions for exchanged data elements. With model-based data engineering, organizations use a common reference model, which offers further clarity and performance improvements. Organizations can use the resulting data to configure mediation services, translating dialects of new or legacy services into a common language for use in the service-oriented architecture.

To support operations with rapidly changing requirements, organizations need service-oriented architectures rather than traditional solutions. These traditional solutions—especially those based on the well-known "waterfall models"—lack flexibility in that they require users to predefine requirements. So long as all requirements are known, such approaches are sufficient. More flexibility is needed, however, if new requirements might arise during the software's use, as when new users have different requirements or the software is applied in a new operational environment. In contrast to traditional system-centric solutions, service-oriented architectures identify, compose, and orchestrate services that fulfill current users' requirements in an ongoing operation.

To enable service composition on the fly, organizations must facilitate *meaningful* semantic data interoperability for information exchange between their services. Although XML extends interoperability capabilities, XML alone is insufficient to cope with semantic data interoperability because it standardizes only a tag set's structure, not its meaning.

Our XML-based approach transfers knowledge of heterogeneous, distributed databases into an XML-based mediation service. Our method extends XML concepts by managing the meaning of tag sets, and thus supporting their mapping. Here, we offer an overview of our approach and its application in the military operations domain. We've published a modeling and simulation-specific application view of our work elsewhere.[1]

Data Engineering

Currently, Web services represent a defined set of industry-supported, open standard technologies that work together to facilitate tag-set-based interoperability between heterogeneous systems, whether within an organization or across the Internet. However, service-oriented architectures' real potential lies in their ability to compose services and enable new functionality compositions that can fulfill users' current—and often changing—requests "on the fly." To accomplish this, information must be exchangeable between all composed services. This exchange entails more than simply exchanging bits and bytes. Services must ensure consistent data interpretation so that services and users possess the same information, knowledge, and—ultimately—awareness. Each service must therefore know:

1089-7801/05/$20.00 © 2005 IEEE

- the data's location,
- the data's meaning and context, and
- which data format the system requires for the data to be useful within respective distributed application services.

Data engineering seeks to locate this information through four primary processes: data administration, management, alignment, and transformation.[2] In XML environments—particularly in Web service architectures—Web-based standards support these data engineering processes. Because each service uses XML to define its information-exchange needs, many translation problems are already solved. When services publish definitions using universal description, discovery, and integration (UDDI) registries, they also directly support data administration. Further, Extensible Stylesheet Language Transformations (XSLT) is an easily applicable data transformation standard. Such standards support the intellectual process of mapping among different information-interpretation structures. Our goal is to establish a coherent framework that combines all of these steps into a new data engineering methodology.

Data Administration

The data administration process manages information exchange among services, including source documentation, format, and the data's validity, fidelity, and credibility. Data administration is therefore part of the service architecture's overall information management process. Applying a general XML policy solves data interchange's technical aspects, such as agreeing on a common format and accessing the data. Furthermore, UDDI registries help users get information about data location.

What's missing in this area are defined tag sets and values to cope with content-specific information descriptions such as data validity, fidelity, and credibility. A well-considered data administration process that's connected with data management simplifies data alignment by ensuring not only that all meaningful information is transferred, but that it's transferred in a meaningful way.

Data Management

Data management is the main intellectual process in the data engineering chain. Data management identifies and describes data elements, and maps equivalent information expressions to each other. Within XML environments, data management is essentially tag-set management. The challenges here are not trivial, and are closely related to problems in heterogeneous, distributed database environments.

In our work on heterogeneous data federations,[3,4] we identified four classes of conflict that data management must solve. These classes are also applicable to semantic XML tag-set management:

- *Semantic conflicts* occur when local schemata concepts must be aggregated or disaggregated, but fail to exactly match (they might overlap or be subsets of each other, for example).
- *Descriptive conflicts* occur when the same concept is described using homonyms, synonyms, or different names, attributes, slot values, and so on.
- *Heterogeneous conflicts* occur when concepts are described using substantially different methodologies.
- *Structural conflicts* occur when the same concept is described using different structures.

Spaccapietra and colleagues concluded that to support efficient data management, we need a generic metadata model comprising only objects and attributes for values and references.[3,4] Their model maps surprisingly well to XML structures.

When the XML schemas to be mapped are relatively simple, data management is easy. Some researchers are already evaluating techniques that automatically generate solutions using intelligent software agents and other technologies.[5] So long as addresses and packing lists must be mapped, these approaches are valuable and should be supported. However, real-life applications—such as those for international military cooperation—are often too complex to be automatically mapped and we need an alternative approach.

The current state of the art in data administration is that experts in both the source and target data models must agree on three types of mapping:

- *Conceptual mapping.* At this level, experts must agree on the data models' conceptual correspondence (an "employee" is a "person," for example). Conceptual mapping lends meaning and validity to the data mapping process; it expresses the intent of data modelers on both sides.
- *Attribute mapping.* This is the next logical step. Once conceptual mapping is complete, the modelers must agree on what attributes reflect identical concepts on both sides ("Social Secu-

rity Number" is "Employee ID," for example). At this level, complex mapping issues (*n* to *m*) are usually resolved.

- *Content mapping*. In most cases, content mapping is erroneously interwoven with attribute mapping. Content mapping expresses the correspondence between attribute values ("<Total Price>" is "<total purchase> + (<state tax>*<total purchase>)"). At the attribute level, we express the relationships between attributes; at the content level, we refine them by defining the mathematical relationships between those attributes.

Data management results in a well-understood and well-documented model of the information that services share. Data management methods are also applicable when additional services are introduced to fulfill new requirements.

Data Alignment

Data alignment ensures that the data to be exchanged exists in the participating systems as an information entity or can be derived from the available data (using aggregation or disaggregation, for example). The data alignment process is crucial to ensuring that the reference model can provide the needed information in a meaningful way.

We can view data alignment as determining the lowest common denominator between two models. If model A has "Employee: {name, age, salary, marital status}" and model B has "Person: {name, date of birth}," the data alignment process will ensure that either model B is extended or that it contains another means of deriving the person's marital status. Model B might, for example, contain a separate table grouping individuals by their marital status: "Status: {single, married, divorced}".

Data alignment compares the producing model's tag sets with every tag set in the target data model. The result is an awareness of gaps—and, hopefully, actions to close them. In the long term, data alignment will help developers more coherently model and understand the application domain's information sphere.

Data Transformation

Data transformation is the technical process of aggregating and disaggregating the embedded systems' information entities to match information exchange requirements, including any needed data format adjustments. In every data model, developers make assumptions on the basis of modeled data and the target intended users. As a result, different models modeling the same data often have different viewpoints depending on the target users. A person in a hospital database, for example, has a different level of detail—or *resolution level*—than a person in an office database.

Recent interface-driven solutions focus almost exclusively on data transformation. In a peer-to-peer effort, individual interfaces between systems transform data without making the data administration, management, and alignment results accessible and reusable for other projects or services. This fact alone shows the clear need for data engineering.

Data transformation must cope with more challenges than simply mapping tags on the basis of one-to-one relations. Generally, the four conflicts we identified above—semantic, descriptive, heterogeneous, and structural—require information aggregation and disaggregation, data restructuring, and so on. This creates reusable and stable solutions only when it's based on engineering principles as established by the application of data management.

Model-Based Data Engineering

For service-oriented architectures to support real-world operations, we must solve two apparently conflicting situations:

- To support a user with the required functionality, independently developed and published services providing this functionality must be composed and orchestrated in meaningful ways.
- The data structures describing real-world operations are too complicated to be handled, managed, and mapped automatically.

In other words, we must manage the information that services exchange to ensure semantic consistency—and do so without knowing the services at definition and implementation time. Traditionally, such efforts have been limited to individually designed point-to-point interfaces. As various researchers in systems' interoperability have pointed out, the mapping problem is an n^2 problem: Whenever a new system is introduced, it must be mapped to every potential partner. If all participating systems use a common reference model, this effort is theoretically reduced to an *n* problem: The new system must align only with the reference model, rather than each participating partner. When developers use a reference model

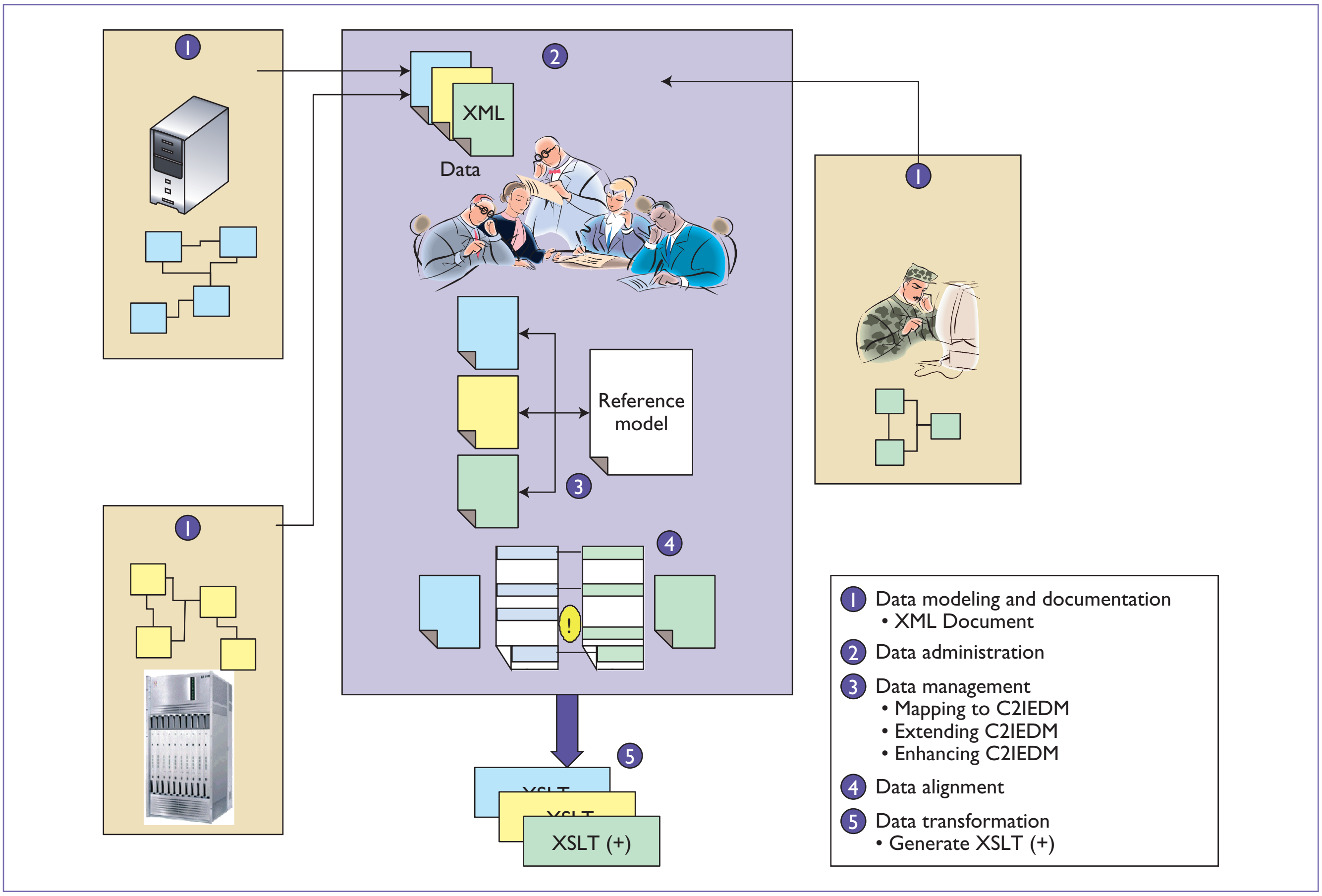

Figure 1. The integration phase of model-based data engineering. Currently, the data administrators and managers often conduct the displayed activities implicitly, with no alignment. One goal of data engineering is to align and orchestrate these subdomains to increase the effectiveness and efficiency of composable services.

to manage data, they gradually create a common information hub that increases in size with each new system.

To apply data engineering to a broad family of services, we need a general approach that uses properties to describe data in context using the following elements:

- *Property values* are the allowed values for a specifying characteristic, such as enumerations "blue, white, red" for the property "color."
- *Properties* specify characteristic values, such as attributes in the relational model.
- *Propertied concepts* collect and structure definitions for a specific entity, such as tables in the relational model.
- *Associated concepts* are semantic entities that describe data in a broader context, such as views in the relational model. They often reflect the organization's higher business objects, such as plans or workflows.

Our model-based data engineering (MBDE) approach uses this information model to cope with the information in the reference data model. In practice, a service's information exchange requirements—the required data and its required structure—must be mapped to those reference-model data sets that have the same meaning.

In MBDE, the reference model essentially serves as the common language. If a model wants to use this language but has a higher resolution, MBDE's extension and enhancement rules allow language refinement to handle this new information exchange requirement. Technically, XML can capture the models and their mapping results, but not internal details on how services handle information. MBDE's reference model is aimed at external information exchange, and doesn't force the service implementation to use special methods or structures, so long as its data aligns with the information exchange requirements

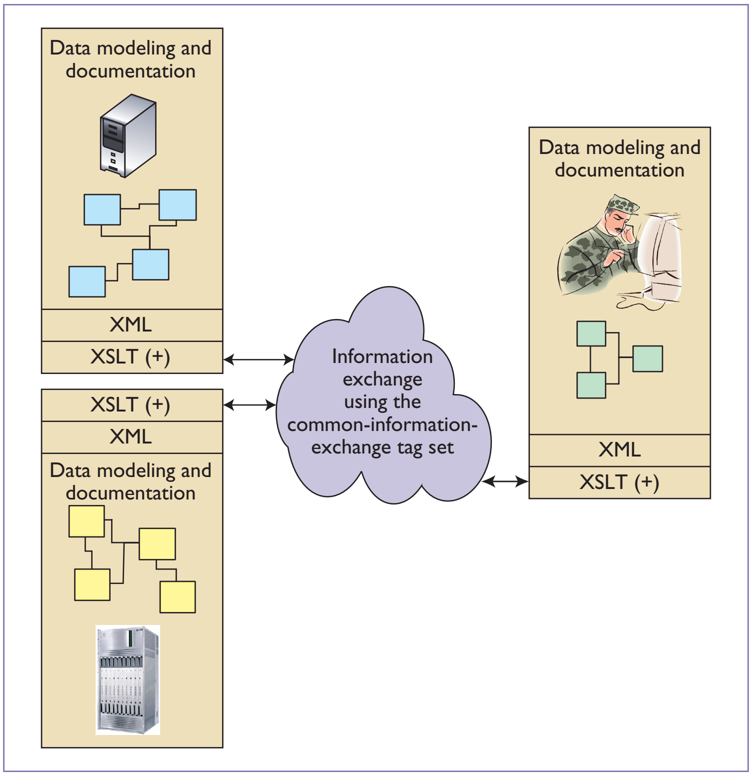
Data modeling and documentation
XML
XSLT (+)
XSLT (+)
XML
Data modeling and documentation
Information exchange using the common-information-exchange tag set
Data modeling and documentation
XML
XSLT (+)

Figure 2. The operational phase of model-based data engineering. XSLT(+) is an enhanced XSLT layer that permits on-the-fly service composition.

Using MBDE to Integrate XML-Based Services

At Old Dominion University's Virginia Modeling Analysis and Simulation Center (VMASC), we used our approach to integrate modeling and simulation services into military service-oriented architectures.[6] As an MBDE reference model, we used the command and control information exchange data model (C2IEDM), which was developed by the North Atlantic Treaty Organization (NATO) for information exchange within multinational military operations. Using a commercial mapping tool, we mapped the XML tag sets of the integrated systems' services to C2IEDM's XML tag sets. To enhance or extend C2IEDM, we followed the rules of data MBDE specified earlier and documented the change to facilitate alignment with other partners and potential standardization. This mapping process produced a mediation service that speaks the services' dialect to one side and C2IEDM's dialect to the other.

Figure 1 shows the five-step MBDE integration process:

1. Data modeling and documentation: Service developers or providers from participating systems use XML to model and document data and document their interfaces.
2. Data administration: Data administrators in participating organizations collect and store all XML documents using UDDI or alternatives.
3. Data management: The organization's data management agency uses the common reference-information-exchange data model to unambiguously define all data elements' meaning, resulting in a mapping of the target XML tag set to the standardized XML tag set.
4. Data alignment: The organization's data management agency compares the data deliverers' supported tag sets with the requested data consumer's tag sets. If all requested tag sets can be delivered, there's no problem; otherwise, the data can't be obtained from that source.
5. Data transformation: Based on the results, service providers can automatically document the mapping as enhanced XSLT documents. Additionally, this results in configuration files for software layers, and hence eliminates the source of ambiguous interpretation of documents by developers.

In real-life examples,[6] however, a simple one-to-one mapping is not always possible. Much more complicated data manipulations are required, often going beyond XSLT's power. In this case, developers need alternative layers based on software, such as that provided by various mapping tools. As Figure 2 shows, this creates a software layer—an enhanced XSLT layer referred to as XSLT(+)—that permits on-the-fly service composition during the operational phase.

In our project, we applied data engineering ideas as follows. In the integration phase's first step, we worked with partners to produce XML definitions of the participating systems' interfaces. From these specifications, many commercial products can help developers easily create a push/pull-oriented Web service design for information exchange. In our case, we used the Altova Suite, but alternatives are applicable as well. We based our mappings on extensive research of our industry partners and significant expertise in the application domain's subject matter. In step 5, we created a layer that speaks native XML dialect and maps it to the MBDE reference model's common information tag sets. The systems can therefore communicate in this common language without having to use it internally. Once the five steps are complete, each system can talk to and receive information from the mediation service in its native XML dialect.[1] The resulting integration framework supports Web services for military applications. Although the C2IEDM is currently limited to the military domain, we assume it can be easily extended to several civil military applications, including those for anti-terror operations. This is a topic of current research.

Developers can use this framework as a common reference data model in related application fields that require services composition to enable overarching pattern recognition (as in homeland security applications). These same theoretical concepts might also support a cascading Web services framework connecting various reference data models. While agencies' data engineering processes remain independent, the resulting data mediation services can be agency-oriented or enable highly efficient peer-to-peer results in special cases. In emergencies, for example, first responders need to share information. Police, the national guard, and local health organizations can maintain common information-exchange data models for their own systems, but in emergencies, the unambiguous police tag set must map to the national guard and health organization tag sets, and vice versa. This can occur through a mediation service that translates between the information spaces of all first-responder organizations. As a result, police could use their own information system to coordinate their work with the national guard, as well as request beds in the local hospital.

The methods we present here are technically mature enough to be applied to support communities interested in service-oriented architectures. Initial prototypes have demonstrated their feasibility and efficiency.[7] What is currently missing is the community-wide will to agree to such a common way to do business; cultural gaps, rather than technical gaps, are the main obstacles. In our case, commercial industry partners are increasing their support for our methods, which bodes well for the future of a common path in our domain.

Write for Spotlight

Spotlight focuses on emerging technologies, or new aspects of existing technologies, that will provide the software platforms for Internet applications.

Spotlight articles describe technologies from the perspective of a developer of advanced Web-based applications. Articles should be 2,000 to 3,000 words. Guidelines are at www.computer.org/internet/dept.htm.

To check on a submission's relevance, please contact department editor Siobhán Clarke at siobhan.clarke@cs.tcd.ie.

References

1. A. Tolk, "XML Mediation Services Utilizing Model-Based Data Management," *Proc. IEEE Winter Simulation Conf.*, IEEE CS Press, 2004, pp. 1476–1484.
2. A. Tolk, "Common Data Administration, Data Management, and Data Alignment as a Necessary Requirement for Coupling C4ISR Systems and M&S Systems," *J. Information & Security,* vol. 12, no. 2, 2003, pp. 164–174.
3. S. Spaccapietra, C. Parent, and Y. Dupont, "Model Independent Assertions for Integration of Heterogeneous Schemas," *Very Large Database J.,* vol. 1, no. 1, 1992, pp. 81–126.
4. C. Parent and S. Spaccapietra, "Issues and Approaches of Database Integration," *Comm. ACM,* vol. 41, no. 5, 1998, pp. 166–178.
5. H. Su, H. Kuno, and E. Rundensteiner, "Automating the Transformation of XML Documents," *Proc. Third Int'l Workshop on Web Information and Data Management*, ACM Press, 2001, pp. 68–75.
6. J.M. Pullen et al., "Using Web Services to Integrate Heterogeneous Simulations in a Grid Environment," *Proc. Workshop on HLA-Based Distributed Simulation on the Grid*, LNCS 3038, Springer-Verlag, 2004, pp. 835–847.
7. A. Tolk et al., "A Layered Web Services Architecture to Adapt Legacy Systems to the Command & Control Information Exchange Data Model," to appear in *Proc. ACM European Simulation Interoperability Workshop*, ACM Press, 2005.

Andreas Tolk is senior research scientist at the Virginia Modeling Analysis & Simulation Center (VMASC) of Old Dominion University in Norfolk, Virginia. His research interests include the applicability of open standards for interoperability of distributed simulated operations. He received a PhD in computer science and military operations research from the University of the Federal Armed Forces in Munich, Germany. He is member of the executive committee of the Simulation Interoperability Standards Organization (SISO) and a member of the Society for Modeling and Simulation (SCS). Contact him at atolk@odu.edu.

Saikou Y. Diallo is a research assistant at the Virginia Modeling Analysis & Simulation Center (VMASC) of Old Dominion University in Norfolk, Virginia, where he is a graduate student in the modeling and simulation program. Contact him at sdiallo@odu.edu.