Non-diagrammatic method and multi-representation tool for integrated enterprise architecture and business process engineering

Lev Grigoriev  
Business Engineering Group  
Saint-Petersburg, Russia  
griglev@gmail.com

Dmitry Kudryavtsev  
Saint-Petersburg State Polytechnic University  
Business Engineering Group  
Saint-Petersburg, Russia  
dmitry.ku@gmail.com

Abstract — Nowadays enterprise architecting and business process engineering are almost synonymous to diagramming. Diagrams have many benefits and are sufficient for many situations. But as the number of diagrams and their types grows, they overlap and evolve, then it becomes hard to maintain a collection of interconnected diagrams, even with the help of a common repository. Besides the very nature of enterprise architecting requires a lot of matrices (goals-processes, capabilities-processes, processes - applications...). The paper presents the multi-representation approach for enterprise architecture model development and maintenance. Classifications and matrices are the basis of this approach and support knowledge structuring and integration. The ORG-Master tool joins classifications and matrices with traditional diagram-based technologies. The paper pays special attention to the role of classifications and matrices in integrating business processes into enterprise architecture

Keywords- enterprise architecture tool, business process engineering, enterprise modeling tool, matrices, diagrams, matrix method, multi-representation technology

I. INTRODUCTION

Enterprise architecting and business process engineering are proven approaches for systemic and sustainable organizational development. They are supported by corresponding tools [1, 2, 3]. These tools document business processes, enterprise strategies, business capabilities, organizational structures, information technologies and especially their interaction and dependencies. Enterprise architecture management (EAM) and business process analysis (BPA) tools not only capture relevant information, but also process this information, e.g. using reports, visualizations or applying analytical methods. The model development interface is the most obvious part of an EAM/BPA tool. It is the interface used to design, build, maintain and often manipulate the models that make up the architecture.

Generally, models are built and maintained graphically using node-link diagrams (a type of representation scheme, that depicts elements or parameters of interest as nodes, with arrows to indicate the connections). The tool’s model development interface may also use textual interfaces to allow additional information to be appended to the graphical models. Diagrams have many benefits and are sufficient for many situations in management [4, 5]. Diagrams work well for enterprise architecture views, which express the EA from the perspective of specific concerns and stakeholders (in terms of [6]). They are also an essential result of any EAM initiative and are provided to different stakeholders.

Unfortunately diagrams are not so good when there is a need to develop and maintain a holistic and consistent enterprise architecture model, which integrates a system of partial views, especially under the following conditions: evolving large-scale models, different and overlapping diagram types, many diagram instances. Chief architects/modelers, who support a whole system of diagrams, may benefit from matrix-based representation. There are two papers [7, 8] that compared the representation power of matrix and node-link diagrams. Ghoniem et al. [7] showed that matrices outperform node-link diagrams for large or dense graphs in several low-level reading tasks, except path finding. This difference is supported by user study experiments conducted by Keller et al., as they found that node-link diagrams offer better visual representation for small, uncomplicated entities, but they are a complex form of representation for large systems and propagation across systems [8]. Besides, two such researchers, Bidgood and Jelley [9], gave compelling evidence that a matrix editor is needed to combine processes with information architecture. In their case study, they used a spreadsheet application as a makeshift matrix editor, putting business processes on one axis and business data on the other, and entering in the relevant cells values indicating whether the process created, read or updated the data.

The paper presents multi-representation technology, called ORG-Master, which was initially designed for business architecture engineering and seamlessly integrates complex business process models into enterprise architecture. Since classifications and matrices are the basis of this technology, the next chapter provides an overview of matrix-based methods. Chapter III explains the overall ORM-Master modeling approach. Chapter IV describes metamodelling language of the suggested tool. Chapter V describes the tool, its matrix editing functionality and general architecture (incl. integration with diagrammatic modeling). The role of matrices for business process integration into EA is described in chapter VI.
II. RELATED WORK

A. History of matrix methods

Kelly [10] provided a good overview of matrix methods. The following chapter exposes Kelly’s main results:

Within the field of information systems development, the use of the term ‘matrix’ is somewhat different from that used in mathematics. Both share the concept of a grid of values, whose location can be specified by their coordinates, an enumeration along the horizontal and vertical axes. In ISD, this is then extended by using objects, rather than whole numbers, as each item on an axis. These objects possess properties, and there may be some hierarchical structure above the objects. Each element of the matrix can be any (result of a function on some) item(s) present in the model, and is determined by a function involving the items that exist on the axes for its row and column.

Many early software design methods had or have evolved matrix representations. These largely fell into disuse, the graphical approach being preferred and fashionable at that time. The development of graphical user interfaces further favored the use of the diagrams in CASE tools, and hence they mainly supported the graphical side of these methods. The matrix side is still, however, as useful as it was originally. Technology has now advanced, especially in the area of databases, and can support a single conceptual graph represented as both a diagram and a matrix, so the user can benefit from the matrix representation in just those areas where it is an easier paradigm than graphical diagrams.

Further, in different fields various methods have been developed that rely heavily on matrices: IBM's Business Systems Planning (BSP) [11] for strategic information system planning; Quality Function Deployment (QFD) [12] for product design and total quality management; Design structure matrix (DSM) [13] for system decomposition and integration problems in systems engineering of products, processes, and organizations.

Support for these methods is typically limited to proprietary, fixed-method tools, often produced by the organisation that developed the method. This approach reduces the flexibility of the program and the ability of the user to do things his way to best suit the current contingency.

The uses of matrices in methodologies can be divided into three classes: those methods that use matrices as an integral part, such as IBM's Business Systems Planning [11], those that mainly use diagrams but where the use of matrices as an alternate representation is recognized, such as Two-Entity Data Flow Diagrams [14], and methods for which there is no explicit use of matrices, but for which applications of matrices can profitably be found.

Kelly [10] looks at these methods in more detail.

B. Matrix and table representations in modern EAM tools and metamodeling platforms

Matrices are also actively used in business process engineering and enterprise architecture management. For example see the CRUD-matrix (Create, Read, Update, and Delete), which links behavioral elements with information, or TOGAF-recommended matrices: Actor/Role Matrix, Application/Data Matrix, Application/Function Matrix etc.

Matrix representation is supported in the majority of EAM tools:

- Matrix Editor in ARIS Design Platform [15],
- Matrix Editor in IBM System Architect [16],
- Matrix Manager in Casewise Modeler [17].

Metamodeling platforms also support matrix and tabular methods:

- Matrix View and Tabular View in the ADOxx® Metamodelling Platform [18],
- Matrix and Table Editors in MetaEdit+ Modeler [19].

The main disadvantage of these matrix tools is that they do not support grouping on the axis / hierarchical axis. These tools display only names of the objects – instances of two user-selected types in the repository. This is a crucial limitation, since it doesn’t allow working with large amount of data; long lists cannot be processes by human brain. It is especially harmful because matrices and spreadsheets are often used to deal with long lists and Excel is one of the main tools in many offices.

III. ORG-MASTER MULTI-REPRESENTATION MODELING APPROACH

The ORG-Master modeling approach has originally been conceived in the course of the development of the business engineering toolkit in 1998 [20]. Fig. 1 represents the suggested modeling technology.

Non-diagrammatic method and tool are at the center of this technology. The modeling process starts from knowledge acquisition. Enterprise modelers collect
information about an enterprise from various sources (people's memory, documents etc.), then organize it using diagrams, classifications and matrices. Diagrams can be created in ORG-Master graphical editors, which are based on Microsoft Visio. These artifacts can be discussed and agreed upon with managers (managers can also make models by themselves). This step is standard. Then ORG-Master integrates all the acquired knowledge using classifications (hierarchical lists) and matrices – so called “internal” representation. These integration and complex structuring is typically done by highly qualified modelers; however the resultant integrated model is inappropriate for final users and enterprise stakeholders, so ORG-Master provides capabilities to specify and generate partial views from this model (diagrams, text and table reports), which will suit various concerns of various stakeholders.

IV. CLASSIFICATIONS AND MATRICES: THE ORG-MASTER METAMODELING LANGUAGE

Enterprise models are created using modeling languages, which are reflected in a metamodel. The creation of a metamodel is also done by using a modeling language. This modeling language is called the metamodeling language and it indirectly defines the way of modeling [21]. The basic building blocks of ORG-Master metamodeling language (Fig. 2) are:

**Classification/hierarchical list** - the representation format for entities, hierarchical relationships between them and values for the properties of entities. Hierarchy is the main feature of classifications, and the main relationship types are “part-of”, “subclass of”, “be subordinated to” etc.  

**Matrix** - the representation format for relationships between entities from classifiers. Example relationship types are “perform”, “help achieve”, etc. The major part of matrices links 2 classifiers, but n-ary matrices are also supported. **Inference matrix**, allows to infer implicit knowledge via the transitive property of relationships in several matrices,

**Types descriptions** - specify the object types (classes), relationships and reusable attributes of metamodel, together with their taxonomies.

Fig. 3 represents an example of classification: objects (positions) are listed in the window on the left, and pictograms define their types. The window on the right exhibits the taxonomy of types, which can be assigned to the objects in the current classification (see lower right property window).

Fig. 4 represents an example of matrix, which is visualized in the form of two lists. This matrix shows relationships between classification (hierarchical list) of...
“Functional systems” and “Functional processes”, e.g. the selected position - “Administration and improvement of marketing system” consists of “Process architecture engineering...”, “Functional processes engineering” and other functional processes.

V. THE ORG-MASTER MULTI-REPRESENTATION TOOL

The ORG-Master tool for business architecture engineering provides strong matrix-editing capabilities. ORG-Master includes the following modules: enterprise model editor, reporting and query module, diagram editor and meta-editor (Fig. 5).

Classifications/matrices editing capabilities in the system work together with diagramming (graphical editor, graphical reports), which is a hybrid multi-representation technology.

The main module - enterprise model editor applies classifications and matrices to knowledge acquisition (e.g. via collaboration with managers), structuring and integration. It produces large-scale enterprise models (e.g. some example classifications in real-life models include 10000-20000 elements; corresponding matrices have even more relationships). Reporting and query modules enable specification of structure and format of “external” text and table documents, then generate these documents. This module also includes capabilities for diagram generation based on model transformation (similar to [22]). These capabilities support predefined notations and depend heavily on the metamodel – whether it is for commercial organization, or for public administration, etc. Graphical reporting includes automatic layout, but some complex diagrams require additional manual improvements. The recent project with Moscow city administration [23] includes 7 types of diagrams (mostly node-link); several types of diagrams have about 1000 instances. Besides, ORG-Master includes a MS Visio-based diagram editor, which helps to acquire knowledge using diagrams, transform it into classification/matrix form and back (works with the common base) and enables to create complex diagrams for final users based on the information from the integrated model.
Metamodel development and customization can be done using meta-editor – you can define object types (classes), relationship types and reusable attributes, then assign them to classifications and matrices. This module is being improved now in order to integrate principles of ontology engineering [24, 25] and Domain-Specific Modeling [26, 27]. The current version of meta-editor partially supports import from and export to Web Ontology Language (OWL). Although metamodel is fully customizable, ORG-Master methodology implies standard metamodels for certain types of organizations (commercial company, public administration, city). The next chapter exposes a fragment of commercial company metamodel for illustrative purpose, but a holistic description of metamodels is out of the scope of the current paper.

In order to distribute enterprise modeling artifacts portal can be used, which provides integrated hypertext and stakeholder-oriented representation of the enterprise model. The portal is an addition to modeling tool.

The enterprise model editor, which supports classifications and matrices, provides the following functionalities in editing mode: group together related entities in classifications; show/collapse subelements; sort elements in classifications; 2 modes for matrix visual representation (“matrix as two lists”, “matrix as a grid”), show relationships for the selected element in a matrix (“matrix as two lists” mode), support for directions of relationships, assign attributes to objects in classifications and to relationships in matrices, highlights, work with groups of elements, etc. This functionality provides analytical capabilities at the modeler's fingertips, for example, to see objects without any relationship. Horizontal layout of classifications in a matrix (“matrix as two lists”) enables showing the hierarchical axis and long “real life” names of elements. The original, “matrix as a grid”, can provide an adhoc overview of existing relationships. The main differentiating feature of the suggested technology from the other EAM matrix tools consists of the combination of hierarchical lists with matrices.

VI. ROLE OF CLASSIFICATIONS AND MATRICES IN INTEGRATING BUSINESS PROCESSES INTO ENTERPRISE ARCHITECTURE

The ORG-Master methodology uses classifications and matrices to integrate processes into overall architecture. Fig. 6 represents a fragment of enterprise model, which exposes process integration into business architecture (links with IT architecture are omitted in the example).
VII. CONCLUSION

Diagramming is a predominant approach in business process engineering and enterprise architecting today. Although diagrams are very effective for knowledge acquisition and communication, matrix-based methods outperform them in some situations. Integration of large-scale enterprise models and support for matrix methods (QFD, design structure matrix, etc.) are the examples. The paper described method and tool (ORG-Master technology in total), which unite matrix-based representations with diagramming. ORG-Master uses classifications and matrices for “internal” knowledge representation (for professional modelers) and diagrams for “external” representation (for communication with various stakeholders).

Advantages of the suggested matrix editing technology: provides big picture (context) and details simultaneously; combine model development with model exploration and analysis (e.g. gap analysis: highlighted elements without links in a matrix); compactness; large amount of information can be clearly arranged and manipulated at once. This allows the user to make a better decomposition of the problem area into subsystems.

Application of the matrix-based technology was demonstrated for business processes integration into business architecture.

ACKNOWLEDGMENT

The authors would like to thank their colleagues – Valentina Kislova, Tatiana Ouerdani, Alexey Zablotskiy, Evgenya Markelova and others – for an enormous contribution to the development of the ORG-Master technology. Special thanks to Dmitry Koznov from SPbSU for the useful feedback about the method and tool.

REFERENCES


