Journal of Software Engineering and Applications, **2012, 5, 583-594**
http://dx.doi.org/10.4236/jsea.2012.58067 Published Online August 2012 (http://www.SciRP.org/journal/jsea)

Meta to the Rescue—Augmented Metamodels for Better Deployment of Pre-Packaged Applications

Joseph Gil[1*#], Elad Fein[2], Natalia Razinkov[2]

[1]Department of Computer Science/The Technion, Haifa, Israel; [2]IBM Research, Haifa, Israel.
Email: #yogi@cs.technion.ac.il

Received January 15th, 2012; revised February 29th, 2012; accepted March 14th, 2012

ABSTRACT

Recent estimates indicate that more than half the software market belongs to enterprise applications. One of the greatest challenges in these is in conducting the complex process of adaptation of pre-packaged applications, such as Oracle or SAP, to the organization needs. Although very detailed, structured and well documented methods govern this process, the consulting team implementing the method must spend much manual effort in making sure that the guidelines of the method are followed as intended by the method author. The problem is exacerbated by the diversity of skills and roles of team members, and the many sorts of communications of collaboration that methods prescribe. By enhancing the metamodel in which the methods are defined, we automatically produce a CASE tool (so to speak) for the applications of these methods. Our results are successfully employed in a number of large, ongoing projects with demonstrable, non-meager saving.

Keywords: Model-Driven Software Development; Information Reuse; Packaged Applications; Method Authoring

1. Introduction

Software is code [1]. It is no wonder that much of the work in automating, and rationalizing software development is code-centric. Tools such as Rhapsody[1] offer numerous means for modeling the code, viewing it, restructuring it, etc. But, there is more to software than the perspective that the code is the ultimate target, towards which all tools aim.

In a nutshell, this paper is concerned with providing better means, "CASE Tools" so to speak, not for software development, but rather for the process of *adapting existing software to the client needs*. Specifically we show how better understanding of the "meta modeling" concepts standing behind "business transformation methods" makes it possible to develop what might be called a "universal CASE tool" for these methods.

We now turn to explaining better the terminology used in the above sentence. Section 1.1 explains what we mean in saying "enterprise applications". The crucial process of adaptation is discussed in Section 1.2. Next, Section 1.3 briefly discusses business transformation methods, which govern this process of adaptation. Section 1.4 gives a quick introduction to the meta-model ling layers behind these methods. Our contribution is discussed in Section 1.5.

1.1. Enterprise Applications and Adaptation

With all respect due to traditional CASE tools, we cannot forget software related problems which these tools do not answer. Indeed, there are many applications in which the bulk of the cost does not lie with the code production, but rather in an earlier stage. Specifically, we are concerned with the crucial domain of *enterprise applications* (often referred to in the literature with the misnomer *Enterprise Resource Planning* (ERP) [2], or the more accurate and recent buzzword *Pre-packaged Applications* (PAP)), which according to some estimates occupy more than half of the contemporary software market [3].

To understand the difference between the realm of enterprise applications and other kinds of software, consider the ever so famous *Unified Software Development Process* [4,5] as depicted in **Figure 1**.

This skeletal description of the iterative software development process has nine stages, out of which only one (I1 in **Figure 1(a)**) is dedicated to the issue of understanding the business needs. Furthermore, the estimates on the resource consumption (**Figure 1(b)**), undermines the inception stage, both in terms of duration and resource consumption. With this perspective, it is no

[*]Work done in part while author was on sabbatical leave with IBM Research-Haifa.
[#]Corresponding author.
[1]Rational Rhapsody,

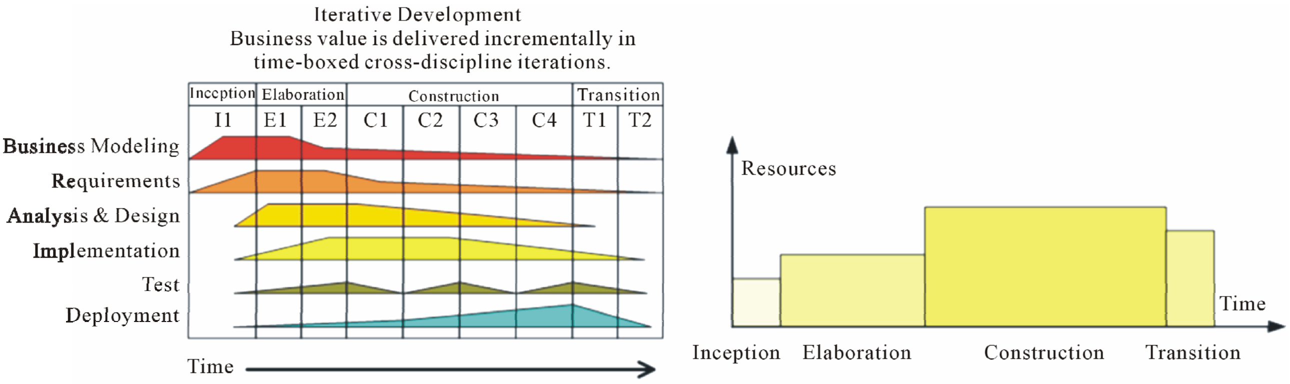

Figure 1. Phases of the unified development method for iterative development. (a) Unified method disciplines (X-axis) and Phases (Y-axis); (b) Typical relative sizes of the four phases of the unified process.

wonder so much effort was invested in developing the wonderful CASE tools we enjoy today.

Let us review some of the reasons why enterprise applications are different than the model portrayed in **Figure 1**.

First is that the complexity of the underlying logic of enterprise applications is different in its nature than the logic complexity of writing (say) a graphic modeling framework—the computation challenges in (say) processing a travel expense report are in taking into account all organizational policies, and these are very different from computational issues in ray tracing. Second, many of these underlying logic operations are recurring—almost every large corporation has some automated policy for managing travel expenses, and these policies are not fundamentally different even in organizations that are of different industries. Third, much of the complication in managing concerns such travel expense reporting (called "business processes" in the lingo), is due to issues such as logging, auditing, transaction management, authorization, etc. Fourth, these concerns and the more or less standard ways in which they are addressed, are even more recurring than the business processes.

As a result, the huge market of enterprise applications is controlled by a small number of "pre-packaged applications" (PAP), also called Enterprise Resource Planning (ERP) systems, providing common business functions and processes across the entire business and its operation. These include the two giants SAP[2] and ORACLE[3], as well as a number of not so dwarfish players such as Microsoft, Computer Associates.

Such software isn't just another "nice to have" piece of technology. When utilized correctly, it becomes the central nervous system of a business. And, ERP systems are often the most important IT projects an enterprise undertakes.

1.2. PAP Adaptation and Deployment

An enterprise wishing to automate its business, or to revise its current automatic processes, must first purchase such a pre-packaged application—a decision which may be costly to make, but often quick to implement. The second, and the infinitely more risky part is the mutual *adaptation* of the purchased software to the business, and of the business to the software. All PAPs need to be configured and customized to meet the unique needs of the business.

In this adaptation phase, individuals with diverse professional skills learn the current mode of operations, configure the purchased software towards it, and make up for the cases in which such configuration is not possible, by modifying the way the enterprise conducts its processes. A polishing stage may follow, in which some, but often not too many lines of code are written for tweaking of the pre-packaged application, to finalize the adaptation process.

In the past, the process to implement ERP typically consumed copious amounts of time, money, and resources. The US army, for example, saw it fit to have a dedicated office for this process (called business transformation office[4]). This reality was particularly troubling to mid-size companies. So while an ERP need may have existed, the process was often prohibitively onerous to consider.

Remedy was found in using dedicated consulting service firms (such as Accenture and IBM Global Business Services), which specialize in this implementation process. Indeed, according to AMR Research, while the total

[2] SAP is the world's largest business software company with more than 47,578 employees at sales and development locations in more than 50 countries worldwide. site: sap.com

[3] Oracle provides the world's most complete, open, and integrated business software and hardware systems to more than 370,000 customers including 100 of the Fortune 100 that represent a variety of sizes and industries in more than 145 countries around the globe. site: oracle.com

[4] http://www.armyobt.army.mil/

ERP software market size is about $34B in 2008, the total ERP service market size reaches about $174B in the same year.

1.3. Business Transformation Methods

An ERP service project, like any consulting service project, is a complex activity which involves months, sometimes, years of effort from hundreds of consultants.

Business Transformation projects differ from software development projects in that there is more of a focus on the business processes, the impact on the organization and the value delivered to the business and the client. Software architecture and implementation typically play a smaller role, especially when the transformation is implemented using a pre-packaged application.

Consulting practices and firms have converged on a number of defined methods for executing projects. Examples include the Oracle Unified method and SAP's ASAP method. These are detailed and structured descriptions of the process of PAP adaptation (sometimes called business transformation).

In theory, the methods provide detailed guidance. Unfortunately, methods were so far only available in the form of static content and not in a way that actively guides consultants. As a result, methods were not followed well.

1.4. Meta Model Layers behind Methods

The *Object Management Group* defines a four level meta-object facility for model-driven engineering. These four levels start at level M0, the kingdom of objects. Objects instantiate elements of level M1, e.g., UML class diagrams. The description of UML resides in level M2. Level M3 is the ultimate level, used by MOF to be build meta-models.

Going down this ladder, we can see how methods are defined. At level M2 we find SPEM, a meta model means for describing methods—the complex protocols for applying business transformation methods. At level M1 we can find various such methods, e.g., SAP's ASAP method for customizing SAP application to the business needs.

The same method may come in different varieties, depending on the business needs and circumstances of applications. These varieties can be thought of as an inheritance tree residing at level M1 (but in a sense, also in M2). At level M0 we find instances of methods, *i.e.*, a process of a specific method application. This process includes also deliverables instances: the concrete documents, test cases, requirements, etc., which are produced in the course of method application.

1.5. Contribution

In a sense, $\mathrm{M_{ATCON}}$ breaks the method content, document and the such, into reusable objects, and enables them to be cataloged and indexed so that these objects can be easily found and reused on subsequent projects. But, there is more to the approach then this, and $\mathrm{M_{ATCON}}$ could be applied even to methods of different fields.

As the title of this paper suggests, the main contribution of this work is in showing that by fully understanding the several levels of meta models offered by MOF, we can create a fully operational support environment, CASE tool if you like, that realizes methods. Specifically, we augment SPEM, allow richer method definition, and show how this richer definition makes translates into a working software environment that support the method.

Also of interest is the way by which we employ existing Jazz tools so that our system involves very little programming, and the way we apply XSLT transformation to create this environment.

It seems as if software engineering is becoming less and less about engineering software, and that the focus shifts to modeling business processes and customize existing software. This work confirms this perspective. We address a software engineering problem, incidentally, the difficult problems of customization, with little programming but with lots of customization of existing tools. Moreover, we do so by (meta) modeling of processes.

On the more practical side, the application of Matcon makes methods operational (the method is "enacted"). This enactment makes it possible for clients, PAP consumers, to stay in better control and risk manage the process of business transformation.

Conversely, our enactment makes it easier for PAP suppliers (particularly IBM GBS) to provide better service, allow the consulting team work in better cooperation, and use the environment we produce to save best practices discovered in the consultation process in a reusable form, void of client specific information, and further easily reuse these best practices. The consultation.

Following concrete estimates on substantial saving in factors such as training costs and 20% - 30% improvement in productivity, and a positive results in one of the world's largest SAP project $\mathrm{M_{ATCON}}$'s output was embraced by IBM GPS, e.g., it was decided that it will be employed in all forthcoming US Oracle projects.

We believe that this success is due to the collaborative, concurrent, and information centric, capabilities of the software we produce from the method definition. These makes it possible for consultants to generate better, more cohesive final documentation, to be fed into the next phase of pre-packaged applications deployment—software development or customization.

We are not aware of much previous work done along similar directions. However, there is work done about SPEM model enactment by transforming SPEM model, e.g., to XPDL model which can be executed by workflow engines, such as Shark [6]. There is also work done about executable SPEM where SPEM meta model is extended to support process enactment [7]. However, these model the process implied by the method, rather than the content, which as we shall see, gives concrete benefits to both customers and suppliers.

Our work on automatic generation of the consultation environment is in the fashion of existing state of the art work on automated model-driven development processes [8].

Outline. *The remainder of this article is organized as follows. Section* 2 *describes our meta modeling approach which stands behind the MATCon project. A brief description of the implementation and techniques used for abstracting out dependencies on the pre-packaged application are the subject of Section* 3. *Section* 4 *describes our evaluation and the resources saved by using MATCon. Section* 5 *concludes.*

2. Meta Modeling

The *Software* (*and Systems*) *Process Engineering Metamodel* (SPEM)[5] specification, is the de facto standard for the definition of what the modeling community calls a *method* (sometimes called *process*). In the context of SPEM, a method is typically a description of a development process, by which an organization generates a software product, but a method can also mean a description of processes other than development, e.g., business consulting. Examples of methods include, modern development processes such as agile[6] and scrum [9] as well more traditional waterfall [10] and iterative [11].

The kinds of methods we tend to think about, are software development methods, but a development method is not limited to software. For example, there is a very definite method for chemical development [12], and in systems engineering, there is a method [13] for the production of complex systems what is called "system of systems" such as aircrafts.

Now, SPEM provides the conceptual framework for a *method author*, to *compose* and *specify* a new method, or for the maintenance of an existing method. Authors here are longsighted individuals or organizations dedicated personnel in an organization who can lift their observations about organizational processes into a concrete structured definition.

In the object oriented tradition, the SPEM framework boils down to a *meta model*, while every specific method is an instantiation of this meta model. According to SPEM, a method consists of two kinds of components, which can be thought of as its nouns and verbs:

1) *Method content*, which is an enumeration of the kind of whitepapers, templates, guidelines, principles, best practices, internal procedures, regulations, training material, etc.—the objects, if you will, on which the method operates, and

2) *Method process*, which defines the workflow and the manner by which content is produced and used. Thus, the process is the definition, in several levels of granularity, of the lifecylce of different content.

A definition of a method by SPEM is more often than not of static nature. Indeed, when method authors sell their methods, customers receive catalogs, documentation, examples, etc. The product of a method authoring tools such as RMC[7] and EPF[8], is not a software system that clients can use. It is rather guidance material to be studied and practiced by team members, enforced by managerial policy, and monitored by team leaders.

On occasions, vendors sell with their methods, supporting software tools which aid customers in the difficult task of implementing the method in their respective factories. For example, tool P2 Toolkit[9] is an implementation of the predefined *Prince*2 method for construction of projects templates.

SPEM talks about a slightly deeper level of method implementation. The term *method enactment* refers to the adaptation of software systems to the method, e.g., the adaptation of Rational Team Concert[10] particularly, its resource planning systems, and work backlog tracking systems, to agile and scrum.

M_{ATCON}[11], the approach, modeling aids, and supporting software, strives to do better than simple method enactment. The grand vision is that with completion of a method, a button press will generate a software system that realizes this method in full. We are not there yet. Method authoring is a long and excruciating process. The resultant method definition is complicated, and typically spans hundreds of different kinds of contents, each with its tangled life cycle prescription.

We do not have the means yet for fully automatic translation of these into a software system. But, much of the remaining manual work is routine, and more importantly requires absolutely no programming.

In principle, the approach we offer is

1) To augment SPEM with further meta-modeling capabilities to support enactment at the level we seek; and

[5] Software and Systems Process Engineering Meta-Model Specification OMG standard, 2008.

[6] Manifesto for Agile Software Development, Agile Alliance, 2001.

[7] Rational Method Composer, IBM, http://www-01.ibm.com/software/awdtools/rmc/

[8] Eclipse Process Framework, http://www.eclipse.org/epf/

[9] P2 Toolkit: Prince2 Support Resources and Documentation, http://prince2.privacyresources.org/

[10] Rational Team Concert, IBM, http://www-01.ibm.com/software/awdtools/rtc/

[11] Incidentally, the acronym sounds a bit like the word "recipe" in Hebrew.

2) To augment method authoring tools with the capabilities to instantiate the richer model, *i.e*., the means and editors which will produce an enhanced method, specifically, including

- A data model for "structured deliverables"; and
- An enhanced state machine description of lifecycles for these deliverables—in particular we provide the means for changing the structure of the deliverable in accordance to its state in its own lifecycle and the current state of the project.

The original SPEM allowed a method to define the kind of tool to be used in each task. For example, in OUM, might be recommended that the requirement gathering task is carried out with Doors[12] or RRC[13], while HPQualityCenter[14] or Rational Function Tester[15] are recommended for testing tasks. Another enhancement we offer is that we allow tools can be defined not only for method wide phases, but also for the lifecycle of each deliverable. This makes it possible for the consultant assistant to not only create test cases (which are kind of structured deliverables), but also pump these into Rational Team Concert for tracking and to Rational Quality Manager for further management of the test case.

Currently, we have not implemented yet an extension to the method authoring tools. Instead, our OUM and SAP instantiation of our approach was carried out manually, relying on knowledge extraction sessions with *Subject Matter Experts* (SME)—individuals who are proficient with the method and its application, almost at the level of the original method author. In these sessions, we capture their view and understanding of the deliverables, in the form of structured deliverables, and the relationships among these. Further, these sessions are to extract knowledge of the lifecycle of each of the deliverables and changes to its structure during the lifecycle. The output is an annotated model in Eclipse's EMF (Core)[16] format—from which the process of generating the environment starts.

2.1. Details of the Meta Model

Consultants' documents are complex work products with a lot of text, lists and tables as well as images for flows and diagrams for architectures, etc. In this section, we describe in greater detail how these deliverables are made into "structured deliverables".

Figure 2 depicts the meta-model we use. This meta-model can be instantiated to a specific model for any document artifact. An adaptation of our system to a specific method is carried out by instantiating the meta model for all types of document artifacts that the method defines.

A principal entity in the figure is Method Deliverable, which represents a sort of deliverables of a method in a raw, unstructured form, e.g., the "conversion strategy document" sort of documents, which is part of the Oracle Unified Method. In contrast, stands the Structured Deliverable entity, which serves a similar purpose, except that it captures commonalities of deliverables of the same sort. For example, the one-to-one association link between these two manifests our desire to replace as much as possible entities of the sort of *Method Deliverable* by *Structured Deliverable* entities.

As shown in the figure, each Structured Deliverable has a number of Sections. First of these are the mundane Rich Text Section, Image Section, and Document Management Section (in charge of storing author name, versions, title, etc.). It should not be forgotten that this list occurs at the meta model level; a specific instantiation of Structured Deliverable will typically require that specific, named sections of a certain type should occur. For example, a use case deliverable (which is an instance of Structured Deliverable), will require a section named "use case details" which would be of type Rich Text Section, and in deliverable of type Business Process would require a Business Process Flow Diagram, which is of type Image Section. Form Section serves a similar purpose, aggregating various scalar values and attributes which may be associated with a Structure Deliverable, e.g., a Requirement document, would include a form with fields such as Priority.

The References Section makes it possible to associate arbitrary files and resources with the deliverable, either directly as an attachment, or indirectly as a *Unified Resource Identifier* (URI).

More interesting is the Cross Deliverable Section, which makes it possible to express *m*-to-*n* sort relationships between deliverables as dictated by the specific methods. For example, according to the Oracle Unified Method, Business Process Model deliverables come in two varieties, level 1 and level 2, and each level 1 model must include a number of level 2 models as well as Project deliverables, and will accordingly include at least two Cross Deliverable Sections.

Finally, of particular interest is the Custom Section, which can be thought of as the equivalent in the modeling of deliverables of prototypes as in the Self[17] programming language and dynamic typing. Specifically, an instantiation of any Structured Deliverable, may allow for the consultant who creates a deliverable of this kind

[12] DOORS, IBM, http://www-01.ibm.com/software/awdtools/doors/

[13] Rational Requirements Composer, IBM, http://www-01.ibm.com/software/awdtools/rrc/

[14] HP Quality Center, https://h10078.www1.hp.com/cda/hpms/display/main/hpms

[15] Rational Functional Tester, http://www-01.ibm.com/software/awdtools/tester/functional/index.html

[16] Eclipse Modeling Framework, http://www.eclipse.org/modeling/emf/

[17] Self, http://research.sun.com/self/language.html

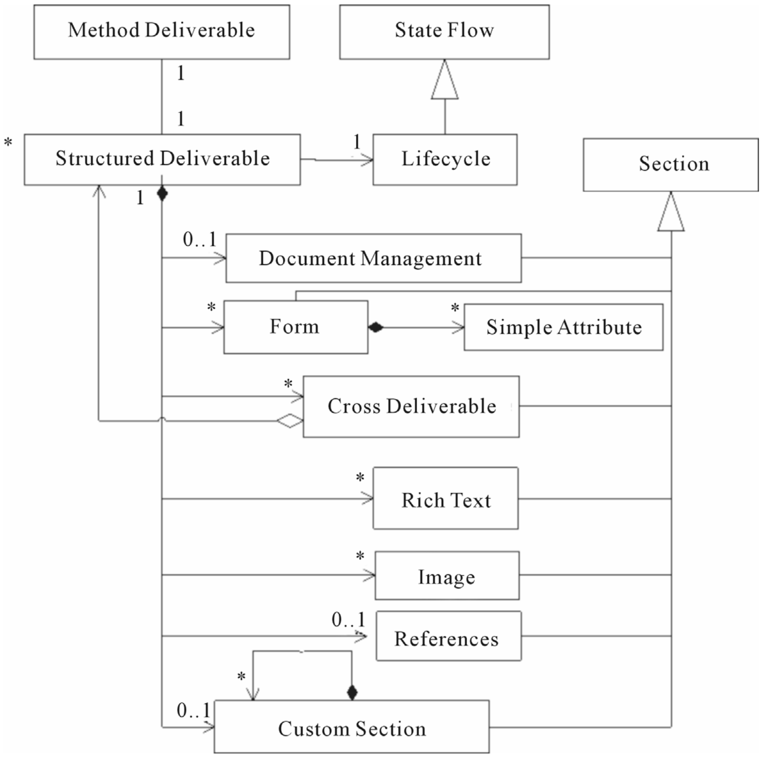

Figure 2. A meta class diagram for the structured deliverable notion.

extra freedom, in that this individual will be able to add any number of Custom Sections which may take a nested tree structure. For example, the Strategy Document and its subkinds, e.g., Application Strategy Document, contain, in addition to their own idiosyncratic list of structured sections, a Custom Section, which allows for a creative, yet partially structured description of the strategy.

2.2. Example: The Key Performance Indicator Deliverable

In this section, we show how **Figure 2** is instantiated for a certain deliverable, specifically the *Key Performance Indicator* (KPI)—one of the most central deliverables of the OUM.

A KPI is a measure of performance specifically related to critical success factor or business goal, which is used by management to steer and improve operations.

We are mostly interested in KPIs that pertain to the value a customer gains by implementing the Oracle solution. For example, introducing an enterprise application to a bank, may include a KPI of the time to process a check deposit. There are also KPIs prescribed by OUM for the consulting phase, e.g., total design time, total cost of deployment, etc.

IBM sell of an enterprise application to a customer begins by comparing the current customer KPI value with the standard or usual value in the industry to which the client belongs. The difference between these two values is what guides the decision of which business processes are to be taken over by the Oracle application and which changes of internal processes are to be implemented.

A customer engagement thus begins with a definition of cross-deliverables, which connect each of the KPIs with the pertinent business processes.

KPIs can be used both during consulting, where they yield current *projections* on customer benefits, and after deployment, times at which they reflect actual benefits achieved.

Using KPIs after deployment is a matter of feeding into this definition numerical values obtained from running the customized Oracle application, as implemented by the programmer according to business process specification written by the consultants as per the OUM. The challenge that $\mathrm{M_{ATCON}}$ meets is that of updating the KPI numbers even during consulting time while some of the business processes are being adjusted.

Figure 3 presents the instantiation of **Figure 2** that we use for upgrading the KPI notion from deliverable to structured deliverable.

In the top left corner of the figure we see class KPI, which is an instantiation of the meta-class Structured Deliverable. To its right we see that a KPI is associated with States, an instantiation of the meta level Lifecycle. The particular instance of States associated with a particular instance of KPI, defines the method phase for

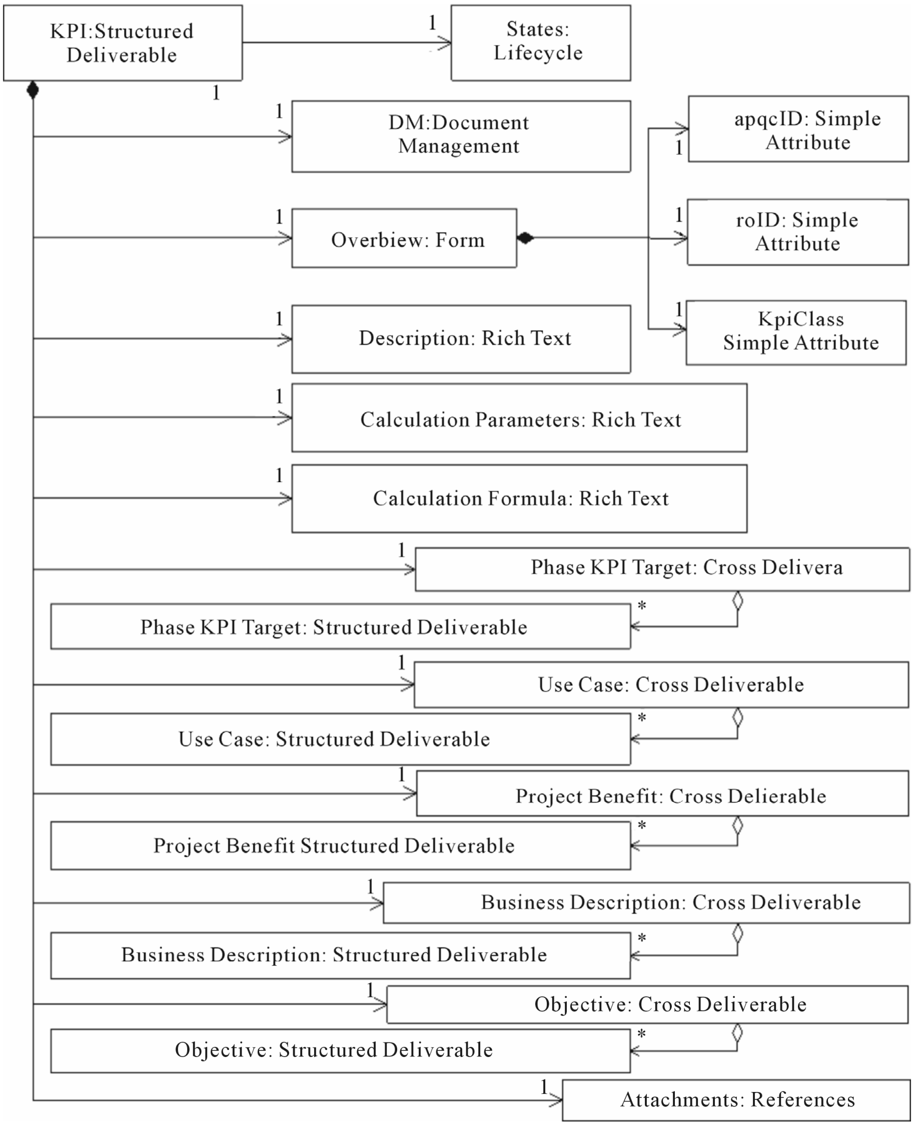
KPI:Structured Deliverable
States: Lifecycle
DM:Document Management
apqcID: Simple Attribute
Overbiew: Form
roID: Simple Attribute
KpiClass Simple Attribute
Description: Rich Text
Calculation Parameters: Rich Text
Calculation Formula: Rich Text
Phase KPI Target: Cross Delivera
Phase KPI Target: Structured Deliverable
Use Case: Cross Deliverable
Use Case: Structured Deliverable
Project Benefit: Cross Delierable
Project Benefit Structured Deliverable
Business Description: Cross Deliverable
Business Description: Structured Deliverable
Objective: Cross Deliverable
Objective: Structured Deliverable
Attachments: References
1
*

Figure 3. A class diagram for the KPI deliverable.

which this KPI applies.

Below these we see that a KPI has a number of sections, all of which are instantiations of appropriate section classes in the meta model. Specifically, the figure tells us that a KPI has these sections:

- DM (an instantiation of Document Management);
- Overview (an instantiation of Form including three instantiations of the Simple Attribute meta class);
- Three rich text sections: Description, Calculation Parameters, and Calculation Formula;
- Attachments (an instantiation of References); and, *most importantly*,
- Five Cross Deliverable sections which connect the KPI to other structured deliverables.

Of special interest are the Phase KPI Target and Project Benefit cross deliverable links. These two links make it possible to automatically update overall project benefit based on the numerical value of the performance indicator and the phase KPI target: different KPIs come to play depending on the phase the project is in. Thus, in each phase, the project benefit is computed based on a different set of KPIs.

Since our meta model is concerned with documents rather than formulae, it cannot represent this automatic computation. Instead, our code generator has a hard coded component which is responsible for generating code for automatic updates of the project benefit out of the KPI and the phase KPI target, regardless of the meta model and models fed into the generator.

2.3. Model Transformation

We now turn into the description of $\mathrm{M}_{\mathrm{ATCON}}$ model transformations. According to Czarnecki and Helsen's taxonomy of model transformations, our approach is of the hybrid nature. There is 1-to-*n* correspondence between source and target elements. Also, our transformation is what is called *structure driven*.

All model transformations we apply are implemented in XSLT[18] using variables, logic and parametrization and XPath[19]. Recall that the model of a specific method and the model of the structured deliverables of the specific method are recorded in an ECORE representation. This representation is the starting points for the application of Matcon. We also note that the ECORE model uses various annotations to guide these transformations.

Figure 4 shows how these inputs are processed, and how a full consultant supportive environment which enacts the method is created as an output of the transformations.

The transformation process starts at the left hand side of the figure, which includes a method (e.g., OUM, ASAP or SOMA) specification in the SPEM format, along with its $\mathrm{M_{ATCON}}$ augments which are recorded in annotated ECORE.

This recorded information is then pumped through a number of largely independent XSLT converters, which produce the various output products at the right hand side of the figure.

In particular, we have the following converters:

- *Model-to-DB converter* which generates a schema of the database and scripts to create it for storing the information captured by consultants while working on their deliverables.
- *Model-to-RESTful services converter* which generates Java classes implementing REST based web services over database to get and post information captured by consultants and stored in database, e.g., a specific KPI and all its parts.
- *Model-to-GUI converter* which automatically generates Java script dojo widgets for information entry editors (for particular structured deliverables) and sections (for particular parts of structured deliverables), e.g., KPIEditor.js and KPIDescriptioRichTextSection.js files.
- *Model-to-External tools Services and Interfaces converter* which generates the Java and Java script code, for example, for implementing integration with RTC, RQM, RAM and Oracle BPA tools.
- *Model-to-Project Management transformation converter* which generates the three parts of our project management engine, namely a dashboard definition, work items definition and project definition.

Later on dashboard definition makes it possible to instantiate dashboards to track projects working in accordance with the method. Work items definition allows instantiating particular work items to be assigned to consultants. Project definition allows instantiating roles, phases and other project relevant data.

- *Model-to-Document Generation Services converter* automatically generates required documents with appropriate styles and content.
- *Model-to-Asset Repository Services and Interfaces converter* generates code which implements various duties required for model-driven asset repository, e.g., contextualized search functionality, harvesting, cleansing, etc. services relevant to particular method assets.

3. Implementation

In this section we shed additional light on the implementation of the $\mathrm{M_{ATCON}}$ approach to produce *IBM's solution workbench for Oracle*.

The Oracle Unified Method (OUM) is Oracle's standards based method. Oracle claims that OUM is rapid, broadly adaptive, business focused and that it includes project and program management framework[20]. Yet, with 7 phases, 105 activities, 347 tasks, 1649 steps, 62 different roles, 135 templates and 194 artifacts, the method is far from being simple.

IBM GBS adapted OUM so as to make it possible for IBM consultants to apply the method more effectively. First, Oracle's method definition, including overview materials, guidelines, templates, tailored work breakdown structure, was reformulated using the *Object Management Group* SPEM specification. Second, IBM augmented the method with information drawn from Rational Unified Method and IBM's own practice and experience in applying the method. The result is called *IBM OUM Methodology*.

The entire process was carried out using the Rational Method Composer (RMC).

The application of the $\mathrm{M_{ATCON}}$ approach was carried out by extracting method and deliverables information from human experts. This knowledge extraction process lasted about three months, and involved a $\mathrm{M_{ATCON}}$ author questioning asset experts, method experts and practice experts, and recording this extracted information in the ECORE format.

Over a hundred deliverables including their lifecycle were thus captured in our meta model representation. Later on we applied the transformations described above to generate a full blown Jazz based environment [14], a global collaborative platform.

We used Rational Team Concert (RTC) (one of Jazz's applications) for the project management engine, including

[18] XSL Transformations (XSLT) Version 1.0, W3C, 1999, http://www.w3.org/TR/xslt

[19] XML Path (XPath) Language Version 1.0, W3C, 1999, http://www.w3.org/TR/xpath

[20] Oracle Full Lifecycle Method for Deploying Oracle-Based Business Solutions, Nov 2009.

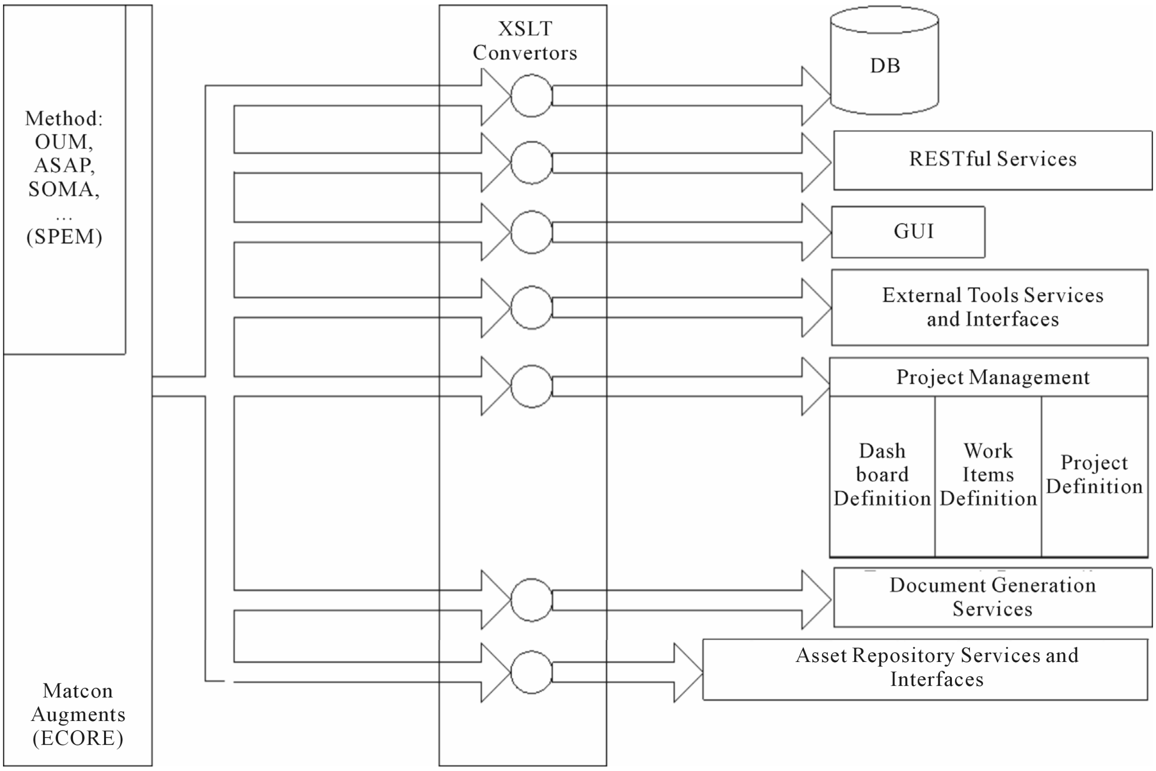

Figure 4. Method to model to environment.

dashboard capabilities and work item management. RTC was also used for implementing a discussion component to the work product deliverables, and for implementation of their approval cycle process.

Rational Asset Manager (RAM) was employed to manage the reuse of the structured deliverables. The APQC/PM process map was used to identify the process we are working on. We used Rational Quality Manager (RQM), while making our Use Cases into Test Cases which can be tested using RQM.

Figure 5 shows an example how one structured delivery—*Business and System Objective* (BSO) is being transformed into a working environment using M_{ATCON}. The figure also shows the rather polished interface for information entry for each model entity. This interface allows consultants to add, delete and edit information and links to other entities defined in the system.

Traversing the figure left to right, top to bottom, we see the inception of the BSO in the Rational Method Composer. Next, this description is enhanced and given an annotated ECORE representation by M_{ATCON}. To generate this representation, knowledge was extracted from the "subject matter expert".

Applying the XSLT converters we generate a full blown *consultant assistant*, which in a web-2.0 collaboration fashion, supports the lifecycle of the BSO, including whatever editing operations are necessary. Working on this artifact automatically creates a Work Item representive in the Project Management environment (RTC).

In the figure we see a specific instance of the BSO, a *General Ledger and Account Payble* document. Furthermore we see that this BSO object connects to Rational Team Concert, which manages the associated deliverable and its approval lifecycle.

Note that the deliverables of this BSO, just as all deliverables are generated automatically, and mapped to the formal deliverables specified by the method for the project. These are the same unstructured documents produce with the absence of M_{ATCON}. This familiarity saves time for the consultants and guarantees that all project documents will be consistent with the template.

4. Evaluation

We would have liked to run a controlled experiment, in which we carry out an enterprise application implementation (say), with and without M_{ATCON}. This wish is futile. Other than the difficulty of keeping all other contributing factors equal, we cannot forget that the introduction of a new enterprise application—a typically $10 million project and circa two years duration [15]—is not only *traumatic* for an organization, it is rather *risky* with surveys indicating failure rate to be between 20% to more the 50% [16]. The bad experience that the mighty HP had [17] that caused it to lose $400 million in the process and to let go of four senior executives is not encouraging. And,

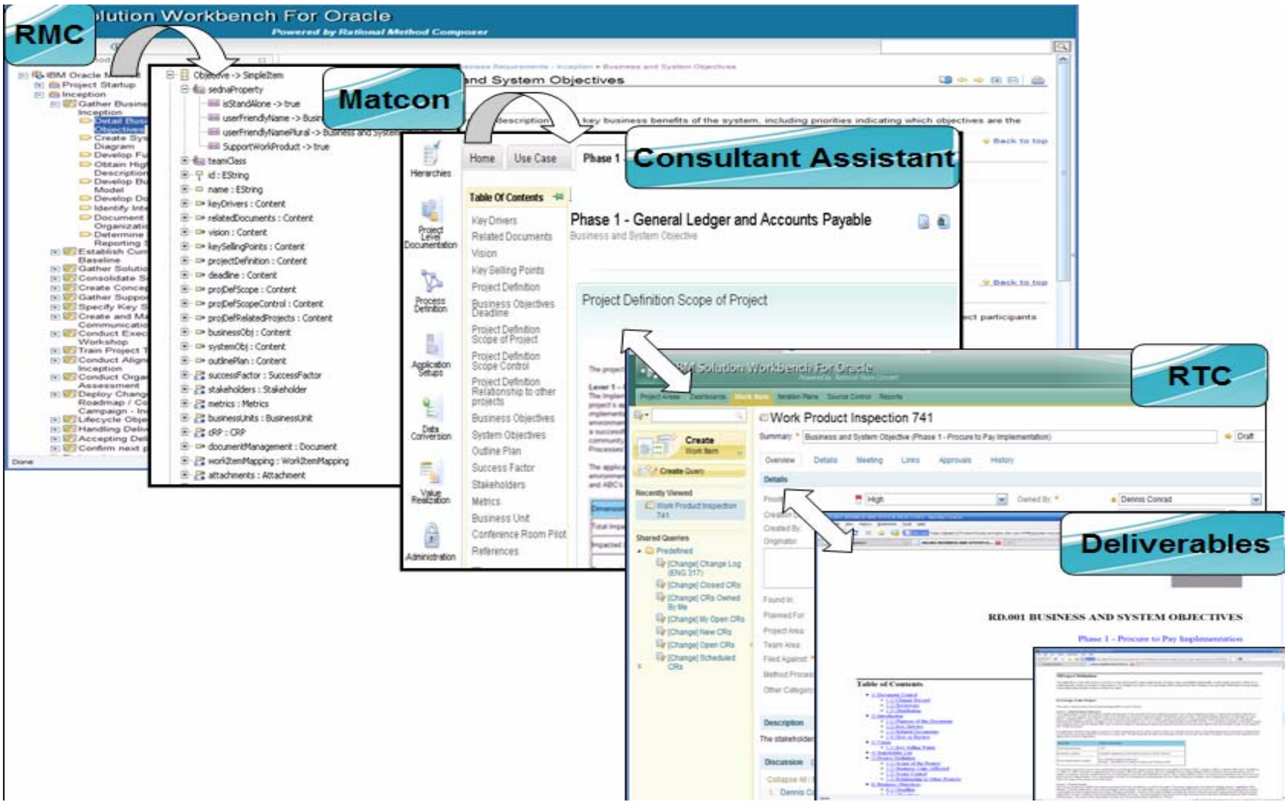
RMC
Solution Workbench For Oracle
Powered by Rational Method Composer
Matcon
Consultant Assistant
Phase 1 - General Ledger and Accounts Payable
Project Definition Scope of Project
RTC
Work Product Inspection 741
Deliverables
RD.001 BUSINESS AND SYSTEM OBJECTIVES
Phase 1 – Procure to Pay Implementation

Figure 5. M$_{\text{ATCON}}$, Method composer and the generation of cosultant assistant.

a recent survey[21] tells us that despite the very structured and very planned process that established methods provide, only 40% of companies had their measurable business benefits expectations achieved at more or less satisfying level. Worse, about 40% of companies surveyed had to forego more than half of their expectations.

Instead, in this section we shall 1) provide data on its *penetration* levels; 2) give measures of the incurred *savings*, at least in the eyes of the suppliers of consultation services; and 3) explain some of the reasons *why* M$_{\text{ATCON}}$ improves the process in a tangible way.

4.1. Penetration and Estimates on Savings

The SAP solution workbench was developed on 2008. The culmination of this project was in a large deployment (probably the largest of its kind) involving 200 consultants, substitute 1100 different IBM applications on a worldwide level introducing one global SAP enterprise solution. This project is at its final stages now.

Development of the Oracle solution workbench commenced at the beginning of 2009. Ten months later it was deployed. Around 150 employees were trained with the system. Currently, it is being used in large scale ERP projects in the US, totaling 45 million dollars employing some 80 consultants.

The use of M$_{\text{ATCON}}$ enabled IBM to apply a conservative 10% reduction in head count and other expenses in these projects. The decision to apply such a reduction was based on internal estimates of various contributing saving factors.

It was estimated that consultants' efficiency will improve by 20% - 30% and that IBM's internal cost to store and manage associated assets will decrease, thanks to 60% cut of redundant or unwanted data.

It was also estimated that the use of standardized solution and environment will lead to 60% - 70% saving in training costs. This savings should allow consultants to focus more on defining business solutions rather than learning tools.

And, by integrating the complete project lifecycle around a *single*, integrated, toolset, rather than multiple tools for each of the tasks in the consultation, it was estimated that an additional 15% work can be pushed to global delivery centers, with consequent cost saving.

Standardization and unification of the tools and assets across all projects and for the entire lifecycle, enhances flexibility, and is expected to permit consultants to easily and quickly move between projects.

Probably thanks to all of these, it was decided that all forthcoming IBM Oracle projects in the United States will be using it. Overall, the plan is to globally employ it

[21] 2010 ERP Report, Panorama Consulting Group, http://panorama-consulting.com/resource-center/2010-erp-report/

in 30% of all projects in 2010 and reach a 70% penentration level by 2012.

4.2. Improvement on Project Success Rates

What are the factors that determine the prospects of success of an ERP project? One may jump to a conclusion that the answer is subject to heated and not very scientific debate. However, Ehie and Madsen [2] carried out a study of hundreds of projects in which eight such factors were identified and then their relative importance was determined by means of linear regression. The most crucial (with ≈0.7 Pearson correlation) factor turned out to be *top management support*.

M_{ATCON} promotes managerial support in two ways. First, the web environment that M_{ATCON} provides allows realtime monitoring of consultants' activity across the ERP implementation. Thus, unlike previous implementations of ERP methods, managers can always stay on top of the progress of the project, guiding it away from joining the sad statistics of more than 50% failures.

Second, and more important is the fact that this monitoring is directly linked with- and updates directly-KPIs—the estimates (or benchmarks) on financial benefits to the organization[22]. The Matcon technique provides a solution to most critical aspect of ERP implementation, that is, in tying benchmarking to implementation, and enabling direct tracking of measurable performance indicators through out the ERP implementation and after it.

We thus have that the use of Matcon enables simultaneously risk management and constant monitoring expected benefits.

According to Ehie and Madsen's scholarly work, the second most important factor detrimental to ERP success is *quality of consulting services*. We identify three ways in which M_{ATCON} should enhance this quality.

1) *Better adherence to method*. Methods are not only structured. They are also very complicated. Each stage within these includes many components, including purpose, relationships, roles, inputs, outputs, inner steps, illustrations, checklists, etc. Without M_{ATCON}, these are mostly words, subject to understanding, interpretation, and good will obedience by humans. M_{ATCON} translates much (but not all) of these definitions into a workbench, that encourages consultants to follow these more accurately, and conversely, use an appropriate method adaptation as necessary. It is a Gartner estimate that a full lifecycle enactment is one of the most important factors to derive the full value from an ERP implementation [3].

2) *Greater reuse and opportunity to offer best practices*. The structured organization of consultation data that M_{ATCON} offers makes it possible, for the first time, for the consulting firm to reuse this data, and let consultants supply clients with best-practice implementation from previous projects.

In the past, IBM's Oracle team spent thousands of man hours to clean a large number assets, so that they could be used on future projects. Internal IBM Global Business Services (GBS) estimates, M_{ATCON} will save 60% - 70% of the effort to harvest clean assets back to the asset repository[23].

3) *More cohesive team work*. Also, this data structuring promotes better full lifecycle application support. That is, more organized communication between the individuals working in different stages of the project: The business analyst team defines the business processes, which are then delivered to the requirement engineers (also called "functional leads"). Without M_{ATCON}, this delivery was by electronic mailing of MS-Word documents. With M_{ATCON} storing these documents and structuring them, the requirement engineers can place electronic ties between the locations in the requirement documents and the relevant components of the business processes documents that initiated their work. Similarly, M_{ATCON} offers a semi-automatic translation of Use Cases to Test Cases and their testing data as generated by the quality assurance team.

4) *More effective collaboration with the client's IT department*. Most ERP projects do not materialize in environments void of any computerization. It is important to explore current IT department knowledge, gather requirements, perspectives and current practice, organize these as per the method guidelines, and reuse it appropriately. Again, M_{ATCON} structured approach is valuable in making these more effective.

We believe these are pivotal to quality. And, as mentioned above, IBM's GBS internal estimates are that overall, the quality of the consultant service is expected to improve by 20% to 30%.

The third factor according to this study is appropriate application of *project management principles*, including scope definition, team setup and responsibility allocation, and performance objectives guided mode of work. It should be clear that M_{ATCON} (together with the method it implements) is instrumental to these.

5. Future Work

We stand still to see the application of M_{ATCON} to more and more methods.

[22] Leading market analysis firms seem to agree that these financial gains ("shareholder value" in the lingo) are indeed tied with these monitoring abilities. In their words (in reference to Matcon): "*... outcomes rather than process...*" (Gartner 2009), "*... link to shareholders value is unique*" (AMR Research, 2009), "*tying benchmarking to implementation*" (Forester, 2009), and "*... tie shareholder value objectives down to the screen and field level within the Oracle applications...*" (TBR, 2009).

[23] Note that GBS is peer to, and different from IBM Research, our division.

Remaining research challenges include better tools for elicitation of knowledge from experts, and enhacement of method editors to support $\mathrm{M}_{\mathrm{ATCON}}$ augmented models. An interesting and intriguing issue remains regarding the proper modeling/meta-modeling of a method together with its specialization modes.

We are in process of collecting actual benchmark numbers based on implementations and usage patterns, and of collecting both clients and suppliers answer to questionnaire. Based on these, it would be interesting to see whether $\mathrm{M}_{\mathrm{ATCON}}$ process can be further improved.

6. Acknowledgements

We wish to thank the team members of the Solution Workbench for Oracle project from IBM Research Pietro Mazzoleni, Vibha Sinha, Senthil Kk Mani, Richard Goodwin, Rakesh Mohan, Biplav Srivastava, Manisha Bhandar, Juhnyoung Lee, Debdoot Mukherjee, Sweefen Goh, Shyh-Kwei Chen, Pankaj Dhoolia, Amit Fisher, Aubrey J Rembert, Jeaha Yang, Mangala Gowri, Rema Ananthanarayanan, our IBM GBS partners Dennis Conrad and Maharshi Desai for great collaborative work on the implementation of the Solution Workbench for Oracle. IBM colleagues Avi Yaeli, Shlomit Shachor, Chris Sibbald, Kai-Uwe Maetzel, Erich Gamma, Peter Haumer, Bingxue Xu helped both fire up and temper our views. Our families provided patience and support.

REFERENCES

[1] M. Weiser, "Source Code!" *IEEE Computer*, 1989.

[2] I. C. Ehie and M. Madsen, "Identifying Critical Issues in Enterprise Resource Planning (ERP) Implementation," *Computers in Industry*, Vol. 56, No. 1, 2005, pp. 545-557. doi:10.1016/j.compind.2005.02.006

[3] D. Ganly, "Life Cycle Guide to Erp and Other Business Application Suite Research," Gartner, 2009, ID Number: G00167163.

[4] I. Jacobson, G. Booch and J. Rumbaugh, "The Unified Software Development Process," Addison-Wesley, New York, 1999.

[5] J. Bosch, "Design and Use of Software Architectures: Adopting and Evolving a Product-Line Approach," Addison-Wesley Professional, New York, 2000.

[6] Y. Feng, "Spem2xpdl: Towards Spem Model Enactment," *The* 2006 *International Conference on Software Engineering Research and Practice*, Las Vegas, 26-29 June 2006.

[7] X. C. R. Bendraou, B. Combemale and M.-P. Gervais, "Definition of an Executable Spem 2.0," 14*th Asia-Pacific Software Engineering Conference*, Nagoya, 5-7 December 2007.

[8] A. Aldazabal, T. Baily, F. Nanclares, A. Sadovykh, C. Hein and T. Ritter, "Automated Model Driven Development Processes," *Proceedings*, *ECMDA* 2008—*Tools and Process Integration Workshop*, Berlin, 9 June 2008.

[9] K. Schwaber, "Scrum Development Process," *Advanced Development Methods*, 1986.

[10] W. W. Royce, "Managing the Development of Large Software Systems," *Proceedings IEEE WESCON*, Los Angeles, 25-28 August 1970, pp. 1-9.

[11] A. Cockburn, "The Impact of Object Orientation on Application Development," *IBM Systems Journal*, Vol. 32, No. 1, 1993, pp. 420-444. doi:10.1147/sj.323.0420

[12] M. Raddatz, M. Schluter, S. Brandt, M. Jarke, T. Grimbach and M. Weck, "Identification and Reuse of Experience Knowledge in Continuous Production Processes," *Proceedings of* 9*th IFAC Symposium on Automated Systems Based on Human Skill and Knowledge*, Nancy, 22-24 May 2006.

[13] J. O. Clark, "System of Systems Engineering and Family of Systems Engineering from a Standards, V-Model, and Dual-V Model Perspective," *Proceedings of* 3*rd Annual IEEE International Systems Conference*, Vancouver, 23-26 March 2009. doi:10.1109/SYSTEMS.2009.4815831

[14] F. Lanubile, C. Ebert, R. Prikladnicki and A. Vizcaino, "Collaboration Tools for Global Software Engineering," *IEEE Software*, Vol. 27, No. 2, 2010, pp. 52-55. doi:10.1109/MS.2010.39

[15] E. Umble and M. Umble, "Avoiding Erp Implementation Failure," *Industrial Management*, Vol. 44, No. 1, 2002, pp. 25-34.

[16] C.-S. Yu, "Causes Influencing the Effectiveness of the Post-Implementation Erp System," *Industrial Management & Data Systems*, Vol. 105, No. 1, 2005, pp. 115-132. doi:10.1108/02635570510575225

[17] M. L. Songini, "Sap Implementation Contributed to hp Shortfall," *Computer World*, August 2004.