Extreme Innovation

Dieter E. Jenz

The gap between business and IT and, hence, the need to significantly improve business/IT alignment have remained on the list of unsolved issues for several years. This unfortunate situation gives rise to an obvious question: Can we just continue on trodden paths and hope for technological advances and more or less smooth evolution, or has the time come to pause for a moment and start thinking in radically new and innovative directions?

To start with, we could boldly attempt to savagely reduce complexity in the problem domain. Given that it is hard to bridge the gap between business and IT, why not try to eliminate one of the two? Since we won't want to eliminate business, what would happen if we eliminated IT from the game altogether?

As a consequence, business analysts, as domain experts, would need to be able to define business processes together with all related artifacts, such as business items (e.g., manufacturing orders, invoices, etc.), business rules, business tasks, and so on. Of course, no one would expect business analysts to produce program code. However, business analysts could describe real-world concepts in the form of models that truly represent business reality and do not suffer from information loss.

Given that IT is left out of the game, how could business process models be executed? By business process models, we refer to the entire business process spectrum, ranging from individual ad hoc human-centric workflow through to highly automated integration-centric processes. This poses a particularly steep hurdle with ad hoc human-centric workflow representing the most critical set of requirements.

Pondering a Provocative Idea

First of all, we strictly need to cater to the requirements of the business analysts who are to work in the modeling space. They need to be able to describe the construction and the operation of the enterprise or parts of it (i.e., business systems) regardless of the implementation, meaning that we need an implementation that is agnostic and undistorted by technological factors.

There are basically two approaches that help with describing the essence of the enterprise that is intended – firstly, the Platform Independent Model (PIM), as presented in the OMG's Model-Driven Architecture (MDA) publications, and, secondly, an ontology-based approach. The latter, Enterprise Ontology, has been described by J. Dietz[i].

The ontological system perspective considers the construction of the enterprise. The enterprise ontology approach is geared to comprehend and design the enterprise in its essential form, totally independent of and abstracted from its actual or possible implementation. The focus is on the essence of enterprise operation, based on common semantics.

DEMO (Design and Engineering Methodology for Organizations), also devised by J. Dietz, is a methodology for constructing enterprise models based on enterprise ontology. Four different aspect models constitute the complete ontological knowledge of an organization. One of these models is the process model, which contains the causal and conditional relationships between transactions that, in loose terms, represent process activities.

Now, how is it possible to proceed to the implementation level in a systematic way? Would it be possible to generate the implementation from a business process model? In concrete terms, would it be possible for a business analyst to model a business process and then execute the model? This is not possible because we need architecture on the enterprise level – that is, Enterprise Architecture.

Eliminating the Gap Between Enterprise Ontology and Enterprise Architecture

Enterprise architecture refers to the function and construction design aspects of the enterprise. Enterprise design is concerned with the business, organizational, informational, and technological arrangements that ultimately determine the actual manifestation of the enterprise.

As of today, the term Enterprise Architecture does not yet have a universally accepted meaning. In the course of time, several enterprise architecture frameworks have emerged, such as the Zachman Framework and The Open Group's Architecture Framework (TOGAF). Many architecture frameworks exist today, and each framework has its own philosophy and focus.

Enterprise architecture forms the bridge between fully implementation-agnostic ontological models and actual implementation. There must be a means to execute business processes; system design guidance is required in the form of enterprise architecture. Ideally, an enterprise architecture plays a pivotal role in the continuous improvement process of an enterprise[ii].

In the model-driven software development world, the model driven architecture (MDA) approach starts with the computation independent model (CIM), then progresses with the platform independent model (PIM), the platform specific model (PSM), and finally arrives at the implementation specific model (ISM). Automated tools would generate artifacts, such as code from platform specific models. Viewed from a different angle, MDA represents a sequential top-down approach. IT experts play a major role in all phases.

How would it be possible to do without IT experts at all? Clearly, we cannot view enterprise ontology and enterprise architecture as separate and sequential activities. If we did, would we not invariably overburden the business analyst with transformation tasks?

A modeling and execution environment that incarnates enterprise architecture comes to the rescue. For the business analyst to remain in control, it is absolutely necessary to shield him/her from the particulars of the technical environment, yet provide him/her with the means to manage an execution environment in non-technical terms. To this effect, a two-pronged approach is necessary: The execution environment must be sophisticated and implement general and self-evident requirements, such as “TCP/IP shall be used as the standard transport protocol.” Secondly, the business analyst must be able to configure the enterprise architecture.

In summing up, enterprise ontology means nothing less than delivering the highest-level constructional model of an enterprise completely into the hands of the business analyst. Configurable enterprise architecture means letting the business analyst control the implementation model in ways she/he can easily understand.

Governance, Risk, and Compliance: Not an Afterthought

Enterprise governance comprises the entire accountability framework of an organization. It represents the organizational competence to continuously exercise guiding authority over enterprise strategy and architecture development, and the subsequent design, implementation, and operation of the enterprise[iii]. As such, enterprise governance provides a frame around enterprise ontology and enterprise architecture.

In the “traditional” software development process, enterprise governance was not considered to have notable significance. Only later on, was its importance duly appreciated. For example, enterprise governance was introduced as an extension of the Rational Unified Process (RUP) in the form of the Enterprise Unified Process (EUP)[iv].

Given the fact that managing business risk and achieving regulatory compliance are among the greatest challenges that enterprises face, business analysts must be able to directly implement governance, risk, and compliance management. This, however, poses a steep challenge.

Business Components: A Powerful Concept

How is it possible to reconcile all of these concepts – enterprise ontology, enterprise architecture, governance, risk, and compliance (GRC) – in such a way that a business analyst may master business process modeling, execution, monitoring, and analysis?

A truly innovative approach would encapsulate business and technical aspects – business components as building blocks. In such a scenario, all business components would rely on a set of common principles. For example, every component would be associated with a type (e.g., business activity, business rule, business item, service level agreement [SLA], key performance indicator [KPI], etc.), and every business component would be considered a "first-class citizen," marking a differentiating factor compared with other concepts. In practical terms, this means that even a property (e.g., "City") would be represented as a business component. This forms a stark contrast to object models, where object properties are not "first-class citizens." In addition, every business component is associated with the lifecycle processes that govern its creation, modification, and deletion. Metaphorically speaking, the component type is equivalent to the shape of a building block.

Business components have the following major properties:

- Composable: Business components can be composed to form aggregate components of arbitrary complexity. An aggregate component may represent a business service, a business process, an organization unit, etc.
- Reusable: Business components are inherently reusable. Since business components represent implementation-agnostic models, the potential for reuse is much higher compared to business components expressed as implementation-specific program code.
- Configurable: The implementation behavior of business components can be altered without having to write code.

Business process modeling becomes an exercise of combining and aggregating business components. A business process would represent a business component hierarchy.

Component composition realizes the composition paradigm, advocated by Gartner. By their definition, compositions are orchestrated assemblies of data, processes and services[v]. Solutions are created by assembling and orchestrating independent parts. Business components would represent such independent parts.

Viewed from a different perspective, aggregate business components fit the dynamic business applications concept propagated by Forrester[vi]. Per Forrester's definition, a dynamic business application is "a software system that embodies a business process and is built for change, adaptable to business context, and information rich." As such, a business component represents an executable dynamic business application at the model level, irrespective of whether it is an aggregate or an atomic component.

As a consequence, enterprise ontology, enterprise architecture, and enterprise governance are no longer separated but fully integrated. Therefore, there is no need to deal with a slew of different models, such as use case models, business process models, object models, etc. In contrast, it is possible to extract views from a business component. The ontological view is of particular interest for the business analyst, since it provides an understanding of the organization's construction and operation in a completely implementation-independent way. Other views, such as a governance view and a security view, are also possible. Views are dynamic, virtual models, computed or collated from information in the repository.

Holistic Business Process Management

A holistic business process management (BPM) approach must address three major aspects: the spectrum of business process types, the BPM lifecycle, and the real-world process-modeling imperative.

Business Process Types

Integration-centric business processes typically rely on asynchronous integration over a message bus. Solicited and unsolicited events and exceptions require appropriate features to handle them. Events and exceptions can be anticipated in a process model, and all kinds of data transformation mappings can be defined at modeling time.

The human-centric processes segment covers a broad spectrum, ranging from unstructured processes, characterized by users collaborating at runtime in an ad hoc fashion, to structured production workflow with its well-defined flow of activities based on business rules.

Differences in process types impact implementation, not modeling. For example, the Business Process Modeling Language (BPMN) spans all types of business processes and can be used by human-centric and integration-centric Business Process Management Systems (BPMS) alike. However, each BPMS has its particular strengths and does not cover the full territory. So, BPMS A might be a good fit for integration-centric processes, while BPMS B might support human-centric processes particularly well.

The BPM Lifecycle

Business process management activities can be grouped into six phases that form the BPM lifecycle: process strategy development, process modeling, execution, monitoring, optimization, and continual service improvement. Process strategy development and continual process improvement can be considered process lifecycle governance elements, while the other phases are operational elements in the process lifecycle. The BPM lifecycle processes are predominantly collaborative and ad hoc.

Real-World Business Process Modeling

As of today, process modeling indirectly involves creating and maintaining multiple purpose-specific diagrams. Model elements, such as process activities, process events, business rules, and business items are distributed across multiple diagrams, such as process diagrams and class diagrams.

A process diagram is expected to provide a representation of the full process context. However, business process languages generally fall short of this requirement. For example, how would one define and associate a service level agreement or a key performance indicator (KPI) with a process activity or a business process? BPMN does not provide specific attributes for setting SLA or KPI information that could be used later for business activity monitoring. However, some BPM tools do have the proprietary extensions and capabilities necessary to capture and store SLA or KPI information.

A Sea Change in BPM

As of today, holistic BPM is still far from reality. While the above mentioned major requirements are anything but new, the slow pace of progress is testament to the fact that holistic BPM is really hard to accomplish on conventional paths.

The business component based approach is indeed very promising in that it is flexible and extensible. Therefore, for example, a SLA is represented by a business component, which has a SLA-specific life-cycle process, a set of SLA-specific properties, etc., In contrast to some BPM tools, a SLA is not specified in unstructured textual form but is machine-interpretable and can thus be automatically checked for violation during process execution already.

Combining business components lets the business analyst arrive at a process model that exactly represents reality. The process model incorporates enterprise ontology, configurable architecture, governance, risk, and compliance. Since the model is entirely machine-interpretable, it is directly executable.

With the business components based concept, case management is no longer an issue. The essential distinguishing characteristic of case management is the "unstructured" progression of a case from initiation to its final state[vii]. The flow logic of such dynamic processes cannot be expressed in a process diagram in advance. Only at runtime would a user determine by human judgment, external events, and business rules the activities to be performed, and in what sequence. As a practical example, forming and formulating a service strategy, as described in the ITIL Service Management Model, falls into the case management category. As a general rule, the lower the level of abstraction and the higher the degree of complexity of a process, the bigger the chance that you end up realizing that it is not reasonable to model a business process in detail[viii]. So, it will come as no surprise that only roughly 20% of all business processes qualify as structured business processes.

A new Kind of Modeling and Execution Environment

Obviously, a holistic BPM approach requires a powerful and comprehensive runtime and tooling environment that spans the entire BPM lifecycle. A graphical interface is needed which allows the user to perform all lifecycle tasks, such as modeling, executing, and monitoring business processes.

To effectively support the full spectrum of lifecycle tasks, the user workplace must eliminate the boundaries between model (i.e., the blueprint or template) and instance (i.e., an "object" created by executing a model). As a consequence, in the workplace the user has full insight into the business. The current separation of model and data, as manifest in today's BPMS and application systems, becomes a relic from the past.

Models are executed by an execution engine, which, in technical terms, is represented by an interpreter that is capable of translating models into executable code on-the-fly. In addition, the execution engine performs transformations with the result that the gap between enterprise architecture and IT architecture is bridged. The execution engine implements numerous requirements, such as, for example, "A business component shall be able to expose its state at any time," " Access to business components shall be based on authentication and role-based authorization," or "TCP/IP shall be the standard transport protocol."

Knowledge-Driven Enterprise Engineering: A Bold Step Forward

Enterprise engineering is about the analysis, design, and implementation of enterprises[ix]. It is based on the paradigm that, in general, enterprises are purposefully designed systems. As such, an enterprise can be (re)designed in a systematic and controlled way. Enterprise ontology and enterprise architecture underpin enterprise engineering.

The enterprise engineering process is compact and easy to perform. More or less, enterprise engineering means creating and combining business components, which requires just one role, the role of the business analyst. There is no predetermined order of progression to modeling and combining business components, meaning that the top-down, bottom-up, and meet-in-the-middle approaches can be used in parallel.

The conventional software development process, as it stands today, is overly complex and requires multiple roles and expert knowledge. As an example, a software development process framework, such as the IBM Rational Unified Process (RUP), identifies and distinguishes some 20 different roles. While the roles actually needed depend on project type and size, complexity does remain a problem.

However, to be fair, a business analyst would need sufficient training to get thoroughly familiar with all of the concepts playing a role in business process management. Still, compared to getting proficient in a software development framework, enterprise engineering proves much less costly.

As a knowledge worker, the business analyst works on the modeling level at all times. There is no need to use computer-interpretable languages, such as domain-specific languages. The composition approach is inherently knowledge-driven, which is why the term "knowledge-driven enterprise engineering" (KDEE) is profoundly justified.

In the end, modeling business processes by combining and aggregating business components is a risk-free venture. For example, it is always possible to generate a process model in BPMN syntax, simply because business components encapsulate enterprise ontology, and thus represents a superset of a process model expressed in BPMN.

In some respects, KDEE is a departure from the stepwise refinement approach. The business analyst's focus is on the business component model, which is the only model type where the business analyst can create, add, modify, and remove business components. Therefore, there is absolutely no need to perform model transformations, which, in contrast, is a characteristic of the conventional software development process. Although the pressing issue concerning the loss of semantics during model transformation is high on the agenda of standardization bodies, it will remain a challenging task to establish unified semantics across multiple models. Unlike the conventional software development process, KDEE is not concerned with information loss and ambiguous semantics.

While the business component model is the only modifiable model, the business analyst can request the dynamic generation of various views, such as the ontological process model view, for presentation in the graphical workplace. However, model views are read-only.

KDEE is comparatively simple and straightforward. No IT experts need to be involved. It comes as no surprise, and experience has shown, that the KDEE approach results in a dramatic overall productivity increase and significantly shorter time to market.

Perspectives, Outlook, and Implications

The adoption of innovative approaches has been slow in recent years. For example, the model driven architecture approach to software system development has taken a number of years to evolve and still lacks widespread adoption. Also, the creation of platform independent, executable UML models with UML diagrams has remained a niche practice area. In practice, executable UML tool vendors have added their own proprietary extensions to UML to make executable UML work. As a consequence, we can justify speaking of executable models rather than executable UML.

However, case studies released by software vendors indicate that MDA based software engineering is a viable approach where it meets a fitting application scenario. For example, E2E Technologies Ltd., a Switzerland-based software company, refers to case studies that indicate a dramatic productivity boost that by far exceeds the 1,000% mark. Their product, E2E Bridge, allows for the direct execution of UML, BPMN, and EPC models.

Still, MDA based software engineering does not cover the breadth of BPM. You need to have a model, but, roughly, only some 20% of all business processes fall into the category of structured processes. The other 80% are partially structured processes, collaborative processes, and ad hoc processes. This is where KDEE proves much more agile.

As a consequence, and arguing from a business viewpoint, MDA based software engineering is still IT-centric and appears to be a dead end street. The transition from software engineering to knowledge-driven enterprise engineering seems to be overdue.

Software products that implement the above mentioned holistic BPM approach are still very rare. There is CONTINUITY, a very promising software product developed by Germany based ICS GmbH. CONTINUITY, which was first rolled out to customers two years ago, is characterized as a rapid business implementation suite by ICS. It supports the entire BPM process types spectrum and is based on a powerful business components concept. It comes with a comprehensive set of predefined generic business components based on the TM Forum's Business Process Framework (eTOM) and the Information Framework (SID), constituting a generic enterprise architecture. Referring to customer experiences, ICS reports an impressive 2000%+ productivity

boost across project types. In other terms, you can save 95% and more with regard to time, human resources, and cost.

Relying on a holistic BPM approach, organizations can realize quick wins. The business/IT alignment issue ceases to exist. IT is relegated to software hosting, since software development and maintenance are no longer required. What a brave new world!

---

[i] Dietz, J., Enterprise Ontology, Springer, 2006.

[ii] Op 't Land, M., Proper, E., Waage, M., Cloo, J., Steghuis, C., Enterprise Architecture, Springer, 2009.

[iii] Hoogervorst, J.A.P., Enterprise Governance and Enterprise Architecture, Springer, 2009.

[iv] Ambler, S., Nalbone, J., Vizdos, M., The Enterprise Unified Process: Extending the Rational Unified Process, Prentice Hall PTR, 2005.

[v] Gartner Inc., Composition and BPM Will Change the Game for Business System Design, December 2009.

[vi] Forrester Research Inc., The Dynamic Business Applications Imperative, 2007.

[vii] Bruce Silver Associates, Case Management White Paper, 2009

[viii] cf. Harmon, P., BPTrends, Vol. 7, No. 9, May 12, 2009

[ix] Albani, Barjis, Dietz (Eds.), Advances in Enterprise Engineering III, Springer, 2009.

