# Enterprise Architecture and the Business Rules Life Cycle

Art Tortolero

## Abstract

This article defines and describes the fundamental Business Rule Lifecyle (BRLC) within an advanced logical and physical integration of Enterprise Architecture (EA) including Enterprise Decision Management (EDM).

## Enterprise Architecture (EA)

Enterprise Architecture has a variety of definitions, primarily because the word “enterprise” is overused as a result of the current business environment.. Enterprise Architecture today is the second most important technology after SOA (Service-oriented Architecture) and is a viable and imperative technology for all large companies. Compliance regulations (Sarbanes Oxley) require that companies maintain coherent and detailed transparent financial controls and audits. IBM’s recent acquisition of TeleLogic’s System Architect confirms the importance of Enterprise Architecture in today’s business environment.

As with many other now widely accepted technologies such as CASE (Computer-Aided Software Engineering), object methodology and business rule management systems (BRMS), Enterprise Architecture has undergone growing pains.. Many companies invested heavily but did not receive substantial ROI for a variety of reasons including poor methodologies, tools, management, local politics and inadequately trained staff. In addition, many vendors and consulting companies resisted enterprise architecture because they were not familiar with the technology and were focused on other more circumscribed solution platforms and approaches. But even though there are still authors and professional groups who do not yet appreciate the full value of enterprise architecture, it has been mature for a few years now, will grow substantially and has already been adopted by the United States federal government in their Federal Enterprise Architecture (FEA) reference model that includes Homeland Security, and by, among others, BP, Intel and Volkswagen AG.

Enterprise Architecture is essentially the integration of business and technical architecture using a comprehensive methodology and a shared repository. The business architecture is comprised of the following:

- Business strategy
- Security
- Business rules
- Enterprise Decision Management
- Business object model (BOM) and/or Data Model
- Metadata
- Business process including workflow, orchestration and choreography
- Information Architecture
- Application Architecture
- Analysis and project management tools
- Business-to-technical alignment

The technical architecture, including data and security architectures, consists of:

- Infrastructure – network, hardware, operating systems, enterprise service buses, etc.
- Database, data warehouses, programs, messages
- Design, development and deployment tools and environments
- Integration tools and guidelines

- Standardization

In addition, organizational structure and performance management are also important, though not as often incorporated components of enterprise architecture. The enterprise architecture itself should consist of some current, intermediary or "gap analysis" frameworks, and target frameworks.

The key to effective enterprise architecture methodology is creating useful, up-to-date blueprints or detailed, colored diagrams systematically showing the basic, general views and frameworks and their corresponding detailed component diagrams. The requirements for creating these critical blueprints are the correct enterprise architecture methodology, especially data integrity and change control, robust tools, and effective project management that clearly spells out team tasks, responsibilities and collaborations, particularly between business and technical staff.

The most important principle in creating blueprints is to use pre-existing core object-based enterprise architecture templates. This strategy will prevent the wasteful and very expensive time sink of re-inventing the wheel. The "core" attribute requirement involves the use of an efficient system principle, such as Pareto's 80/20 rule, which applies the concept that within every complex system there is a simpler prototype. The object-based approach offers the best methodology for organizing the data, functions and processes of the enterprise.

This object approach combined with information engineering has now evolved into a discipline called ontology. Ontology is a data model that highlights semantics or meanings that represent a set of concepts within a domain, including their classes or categories, their attributes, relationships, events and transitions. Though the more network-oriented ontology is a promising extension of the object model, there are many essential hierarchical object model features that are not well used. One such example is creating class gerunds by simply converting verbs, in particular, business processes, into verb action nouns by adding "ing.". Gerunds present more useful abstractions within your rule organization.

## Business Rules as Part of the Business Architecture

Though business rules are often defined in their atomic granularity form, i.e.in terms of their declarative, process logic independence, and their if *term/fact*, then *action(s)*, *else other action(s)* basic structure, the most important role of business rules is in their concrete formulation of the business strategy, which we will describe in the next section. This more granular focus is valuable, because by defining rules in this way, you can derive more crisply defined and therefore more testable, measurable, modifiable and reusable rules throughout other company business processes. In enterprise architecture as a whole and in many applications, however, you must still address more general and often more ubiquitous plain business logic which does contain both business rules and procedures interspersed. A practical approach requires the management of both of these business specifications—rules and procedures--in their logical constructions and in their physical development and maintenance,. ideally implemented using a BRMS tool.

## Enterprise Decision Management (EDM)

Enterprise Decision Management (EDM) is a subset of enterprise architecture that encompasses improving the correctness, consistency, changeability, speed and automation of high-volume decisions using business rules management system (BRMS) and business intelligence (BI) for analytics. The tools used are both Rete algorithm-based, inference engines for more unstructured, unknown results with AI-like and expert systems-like applications. Far more commonly used – about 90% - non-Rete, are sequential, known results BRMS tools. An important market driver is the fact that business is still moving faster than current technology in our operations. Decisions in our global, dynamic highly volatile marketplace are more real-time, STP-based (straight-through-processing), complex and sophisticated than our technologies and current operations. EDM also addresses both the management and technology of improving decision making. Though there is much recent attention in EDM to our decision limitations in customer-facing operations, it also is important – sometimes necessary -- to address the role of EDM within the enterprise architecture, especially its methodology and integration with the explosive emergence of BRMS tools and the business rule lifecycle.

The enterprise or application business strategy is the cornerstone for EDM, and the quintessential decision management driver, because it is responsible for creating the overall structure of a very specific, framework blueprint. This framework blueprint must be object-based where the classes are modelled in terms of their strategic and tactical focus – often involving separate diagrams. The operating model may need to consider whether the individual organizational units (companies or departments) will strive for integration, federation or a hybrid. Further considerations are location and other orthogonal factors as well as the panoptic context and discrete delineation of customer targets, products, brands, prices, services and their corresponding marketing, sales, and support matrices. This strategy-focused EDM framework emphasizes the value decision chains that guide the execution foundation and explicit controls, and the business-to-technical integration including deployment and its attendant customer-facing operations. Clearly, a pragmatic, customizable, agile enterprise architecture methodology is indispensable.

## How EA, Business Rules und EDM Work Together

EA includes EDM which includes BRs (business rules). Their successful design and deployment requires a logical coherent and synchronized methodology and a physical, practical, fast and flexible BRMS tool that both are fully specified within the business rule lifecyle that is depicted in the diagram below:

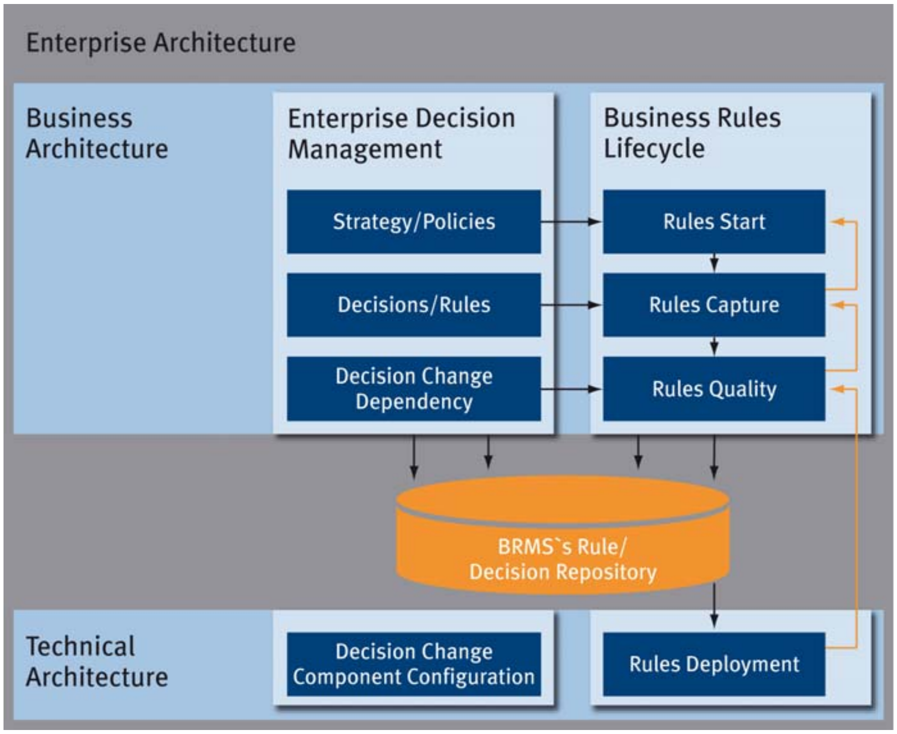

**Figure 1**: **How Enterprise Architecture, Enterprise Decision Management and the Business Rules Lifecycle work together**

Business architecture strategy drives the technology architecture and thus assures that it will not only reduce costs but will also optimize the business operations and opportunities. An enterprise architecture methodology must capture this business strategy in an object blueprint that can viewed and validated by the key business and technical experts. Decisions are important sets of business rules that often contain dependencies. Decisions in business rules change more frequently than business processes or workflow changes and thus engineering to ensure their agility is a fundamental concern. Decisions are the critical success factors, and thus they must be prioritized via automation and illustrated for human manageability from the overall strategy down to the atomic business rules. The quality attributes associated with business rules are even more important in decisions, especially their initial declarative definition, so that they can be more effectively exposed as services with an SOA or ESOA (Enterprise SOA).

Logically, the decision change component configuration will identify all decision rule dependencies and decision owners by role responsibility and currently assigned individuals. It is also necessary to identify all the customer-facing system elements that must be updated, such as user interfaces, reports, customer representative support scripts, IVRs (Interactive Voice Response) systems and the technical system components, such as databases, data warehouses, programs, services and, of course, the BRMS tool, itself, including rule organization, rules, services and test cases. Data is the most stable, central concern, and it is critical to have a transparent data integrity track that pervades all rules stages and their continuous iterations. Business analysts cannot be allowed to introduce new data after a release, unless it is an emergency situation, without first submitting this to the project role manager (whether database administrator, data modeler or rule consultant responsible for maintaining the data model). Likewise, rule sets must have an explicit owner responsible for updating a certain category of rules. Blueprints showing the linkages between the data, rule, and tools help to streamline this critical change control process, for example, the connections between the data model, rule model, user interface screen data, service data and data stores. These blueprints should be displayed in the project “war room” and updated continuously when they get too busy.. Team and task dependencies also are similarly highlighted. The successful management of this decision change component configuration is the foremost requirement for business agility. The ability of a BRMS to visually display the rule change configuration via dependency diagrams also is important.

## How the Business Rule Lifecycle (BRLC) integrates EA, Business Rules and EDM

By extracting the business logic from programs and storing them in a centralized and modifiable BRMS repository (in much the same way data was extracted into database management systems 25 years ago), the BRLC will constitute a separate, more high-level cycle within the standard technical development round-trip engineering lifecyle. While this separation provides tremendous business value, a complete dependency blueprint of its technical touch points and feedback loops is necessary. The blueprint be different for every organization because of the different mix of technologies, role responsibilities and implementation procedure standards. There is a frequently made mistake in this process of starting the business analysis and design but delaying the technical architecture design and development. This mistake is fuelled by the notion that a totally top-down approach is ideal and that, practicality suggests that the technical architecture cannot be initiated until the business analysis is complete. Actually, both a top-down and bottom-up approach are the ideal, because there are, indeed, many time-consuming technical tasks such as development tool evaluations and infrastructure programming that need to be initiated. If they are not, then they will have a negative impact the overall project deliverable schedule.

The BRLC, like enterprise architecture, is not a waterfall either in the macro or micro stages, but a continuous waterwheel of internal cyclical cogs propelled by the twin currents of business and technical detailed droplets and large-scale wave transformations. The BRLC contains the following core stages (see Graphic 1) that are employed by Innovations Software Technology, builders of the premier, sequential Visual Rules BRMS tool:

### Rule Start

Logical:

- Strategic planning
- Scoping, constraints, project and systems requirements
- High-level business process and object (recommended) or data model
- Current, target and intermediate target architectures and applications
- Use general rule templates.

Physical:

- Investigate strategic and tactical business specifications, compliance regulations, systems documentation, memorabilia, etc.
- Select enterprise architecture tool (recommended) or less robust and capable data modeler and visualization tools (for smaller projects)
- A BPM (business process tool), a workflow tool, a BRMS tool, and project and development management tools.

**Rule Capture**

Logical:

- Build high-level business functional decomposition; rule organization and classification
- Rule typology – functional, calculations, exceptions
- Taxonomy – especially formal rule naming conventions; and
- Metadata – data about data such as how to define different versions of service
- Generally identify decisions and their change control configuration.
- Define declarative business rules and model the business logic.
- Identify the decisions, their rule sets and their integration with the business logic within the business processes and workflows.
- Use specific rule templates.
- Capture any analytics or business intelligence (BI), if necessary, and optimize the decision logic accordingly.
- Also specify policy enforcement rules especially for services.

Physical:

- Use visualization tools and BPM/BRMS tools. The key differentiator of the BRMS tool should be the ease and power with which a business user can model, test and change the business rules.
- The data imports and exports especially to spreadsheets where so much business information is kept also are significant.
- The BRMS repository is the heart of the enterprise architecture because it contains the critical decisions and general rules and logic and their physical interface – or "where the rubber hits the road" –via their automated program, database connectivity, test, web services and documentation generation.
- Integrate the BRMS with an analytical tool, if necessary.

**Rule Quality**

Logical:

- Specifically define object-based construction rule organization, rule naming conventions, test cases, simulation, input and output data, dependencies, test reference and result data, regression testing and especially the decision change control configuration.
- Define how the completeness, correctness and consistency of your rules will be tested logically.
- Focus on completing the decision change component configuration including exposing implicit decisions and assumptions.
- Identify dependent analytics, if necessary, and incorporate in the change configuration.

Total quality management (TQM) should be supported by feedback loops including:

- from testing back to the model
- from the release back to the model and
- from audits back to the model.

From release back to the model is especially important since it is a challenge to ensure the correctness and consistency of the rules before deployment.

- There are many other important rule quality attributes that may be necessary to consider in a formal way such as: reliability, efficiency, level of personalization, stability, predictability and transparency.
- Stress the importance of continuous learning, optimizations and innovations.
- What are the values, priorities and their associated mental models captured in company highlights, training, customer and partner presentations and project critical success factors. Check that these business directions are actually in the decision rules.

Physical:

- The BRMS must automatically detect errors and provide measurements of the test coverage including explicit, preferably graphical, variations of the test results and reference data.
- In addition, variations between different archives and versions are necessary.
- The BRMS testing environment should be integrated to expedite these testing requirements and make it easier for business users to detect anomalies including step-by-step graphical debugging views, graphical simulations via deactivated rule branches, and granular test case/rule parameter executions.
- Evaluate the collaboration platform offered by a BRMS in terms of how easy it is for different subject matter experts from different business domains to work with the tool without having training using the tool and how easy it is for the business user and the technical staff to cooperate.

## Rule Deployment

Logical:

- Define where, when and who will get the rule releases.
- Define special customer processing called QoS (Quality of Service), if necessary.
- Define the service level agreement that specifies the workload and response time requirements demanded by your customers, if any
- Define service orchestration in terms of which business processes and specific workflow steps are executed.
- Define the collaboration of different external parties outside of the business processes and their interactions.
- Define event tables for complex event processing.
- Use general business service design patterns and specific rule design patterns to model your business-to-technical architecture.
- Identify legacy integration, migration and modernization req.

Physical:

- The BRMS repository should be explicit, fast and easily accessible in a distributed, 24/7 architecture for large companies.
- In addition, the BRMS tool should provide hot deployment (load, unload, reload) so that the rules can be changed in real-time during production without interruption of the business application.
- Use web services (WS) including, if necessary, WSDL (Web Services Description Language) to define the web service interfaces , BPEL (Business Process Execution Language) to define the workflow, and WS-CDL (Choreography Description Language) to define the collaborations between web services.
- Different BRM, workflow and ESB (Enterprise Service Bus) tools and standards need to be selected.
- Note that services in an SOA do not need to be comprised of web services nor does the ESB.
- An ESB includes invocation, routing, mediation, the web services language executions described above, security, QoS, and administration.

Adapt your enterprise technologies as much as possible to standards without sacrificing any major business functionality or incurring any considerable technical expenses.

Evaluate whether the BRMS tools have easy, open access to mainframe components and can be customized for special functionality.

Within administration, there is another important and emerging technology called BAM (Business Activity Monitoring) which entails monitoring, auditing, logging, alerting and metering. From a web services perspective, the BRMS must be able to work with services both in terms of generating WSDLs that then can become web services and also in terms of calling services from within the rules themselves.

From a deployment perspective, the BRMS also needs to have high performance and scalability capabilities to address the increasing transaction workloads and real-time response time requirements of many business applications. Three important performance considerations for a BRMS are the following:

- whether it needs to do some kind of rule processing during run-time or can instantly fire a rule,
- whether only the rules that need to be executed are processed during run-time,
- and whether it is possible to monitor the statistics of every rule element to determine where the performance spikes are located and develop approaches to optimize these performance-intensive rule nodes.

## Summary

Here are the points I attempted to make in this discussion.The Rule Start, Capture, Quality and Deployment stages must be addressed initially at a high-level and then in a detailed, atomic level. Enterprise architecture methodology is an important technology to fully and successfully design and implement the rule lifecylce. The ability to iterate smoothly must be managed and engineered. The reuse of architecture blueprints and rule templates streamlines the entire process. The interdependent tasks of the business and technical development tracks must be identified to allocate resources proactively to prevent schedule slippage while each track should work on their respective deliverables from the start of the project. This document has concisely described the basic BRLC process, but it is necessary to use a more detailed and rigorous methodology including actual enterprise architecture and decision management blueprints and rule templates.

How a BRMS works with each rule lifecycle must be evaluated during tool selection from a human design as well as technical capability criteria, and then explicitly modelled and implemented during the rule development process, especially the decision rule change component configuration. The combination of enterprise architecture including decision management, SOA, the BRLC and advanced BRMS tools represents a truly mature and historic business and technical productivity and quality quantum leap.

### Author

Art Tortolero is President of Innovations Software Technology. He is a former Senior Enterprise Architect who has pioneered both software engineering using object methodology and enterprise architecture at premier organizations including Bells Labs/Lucent, Ameritech/SBC, AMOCO/BP, Fleet Bank/Bank One, ABN-AMRO, United Airlines, National Computer Center, BroadVision and SunGard.