

THE NEXT BIG STEP IN BUSINESS ANALYSIS: PERFORMING TECHNOLOGY-AGNOSTIC BUSINESS SOLUTION DESIGN

THE EVOLVING BUSINESS-IT RELATIONSHIP

The relationship of IT to the business has evolved significantly over the years. Originally, IT functioned as a tool builder. In that role, areas of the business operations that were tedious and rote were targeted for automation. In order for it to make business sense to build these tools, it would have to cost less to implement and maintain the technology that could do the work than it would if people continued to perform the activities manually. On the human labor side, that cost involved the labor expense as well as the cost of inconsistency in the way people — as opposed to machines — do work. On the technology side, the costs involved the capital cost of the equipment to do the work and the cost of determining how to codify the work into something that equipment could perform.

As the relative expense of automation-supported business behavior moved rapidly down the cost curve, more and more business behavior became eligible for automation. The move to automate paper-based workflows paved the way for automating business processes. But as business processes started to be automated, the nature of the relationship between business and IT began to change. No longer was the business process entirely executed by domain experts with the help of some tools. In this new model, the automation of simple business behaviors (e.g., validating eligibility criteria) moved into the IT implementation. And with this shift, the business side became detached from certain aspects of how the business behaved, because they no longer directly performed all the business work. In turn, IT's role also changed from that of tool builder to that of interpreter. A key consequence was that an additional step was needed to involve IT in many changes that used to be performed purely on the business side.

The addition of this “extra” step to the process of making business changes has laid the basis for a dramatic transformation in the relationship between business and its IT partners. As a tool builder, IT provided new breakthrough technologies that allowed organizations to either perform existing work more effectively or to introduce new differentiated products and services. In its new role, IT has become more of an overhead cost that is tacked onto the execution of run-the-business kinds of changes. Understanding how to change this situation and move IT into a more strategic partnership with the business requires examining in a bit more detail how we got to this point.

This evolution of the business -IT relationship into one where aspects of business behavior are controlled by IT was probably inevitable given the technology and methodologies previously available. When business behavior is rendered into code, it becomes impossible for the domain experts to “read” the actual business behavior. As a result, IT ends up being the custodian and translator of the business behavior captured in the code. This problem has been recognized for

many years, and attempts have been made to solve the problem by making more “business-friendly” technology languages. In fact, COBOL was developed based upon this idea of preserving the ability of domain experts to directly read and understand the business logic contained in IT code. However, these IT language-based approaches to retaining business visibility into the automation solution never succeeded and were probably doomed from the beginning. Why is this the case?

WHY THE REQUIREMENTS-BASED APPROACH FALLS SHORT

Most organizations today have chosen to deal with the loss of business control by basically accepting and embracing it. This approach involves the business developing a set of “requirements” establishing the goals and constraints that define the features the business automation solution must have when it is delivered. At first blush, it might appear that this approach is an effective way of handing control of business behavior back to domain experts, but in fact it fails significantly on several fronts.

Language Can’t Capture the Unknown

The first major problem with the requirements approach is that it is extremely difficult to capture what we want something to be by listing a series of features and characteristics that the delivered solution should have. This is the well-known problem captured in the aphorism “A picture is worth a thousand words.” The message is that language is an inadequate method of documenting a complex system and communicating it to people. That’s why many industries have standardized on various non-language-based techniques such as construction blueprints, electrical diagrams, and CAD systems.

The requirements-based approach is an admission that we have yet to develop and agree upon a set of standards that capture at the business level the details of the business behavior that is expected. Lacking this, organizations instead generate enormous tomes that describe externally verifiable characteristics of what they want. This ends up placing IT into a role not unlike that of the three blind men trying to describe an elephant based solely on what they can feel. Yet this analogy isn’t even a fair one, because in that tale what the blind men are trying to describe is a real cohesive system, the elephant. In most organizations, the requirements approach is more akin to a series of statements describing an animal that a group of people *imagines*. Nobody can really tell what the actual animal looks like or even if the things that are being written to describe it are complete and not contradictory. Because of this approach, IT is forced into the position of trying to create a creature that meets all the requirements while filling in all the gaps and resolving all the inconsistencies and impossibilities.

External Behavior Masks Internal Incoherence

The second major issue with the requirements-based approach is that it only specifies the externally visible characteristics of the delivered automation solution. If we think again of our elephant problem, we might decide that it’s an elephant if it looks like an elephant and acts like an elephant. But what if we dissect the elephant and find that it is full of electronics and hydraulics as opposed to digestive and circulatory systems? Would we then still agree that it is an elephant? If the

external behavior fits the requirements but when we go to make a change we can't understand the way the internals work, then the ability of domain experts to understand and specify changes to the business behavior captured within that system will be severely undermined.

Imagine you have a business rule that says something like "To be an Approved Vendor, a Vendor must have the following characteristics: a completed Dun & Bradstreet profile and a completed commercial terms agreement." Now suppose you asked the IT group that implemented the system containing this rule to show you the "approved vendor" concept. The odds of there being a concept known as "approved vendor" that exists at a single point and corresponds directly to this rule vary greatly from system to system and organization to organization. But what is almost universally true is that there is no expectation that this concept *must* be preserved in the implementation using the same business terms and language. Multiply this omission by the thousands of business concepts and behaviors that go into a typical automation solution, and the failure to preserve the business concepts imposes a significant burden on an organization's ability to understand and therefore modify the executing business behavior.

Thus the requirements-based approach forces IT into a translator role in which it has to explain what the implementation is doing because the internals aren't doing what the domain expert might expect. Think again of our elephant example. If I go to perform surgery on the elephant to repair, say, an intestinal blockage, I would have a very difficult time figuring out what to do if I was told that the elephant had no intestines because the implementation team had found a different but equivalent way of constructing an elephant.

From "Implementing" to "Enabling"

These two issues illustrate the fundamental problem with the requirements-based approach. By encouraging a lack of separation between business design and technical design, the requirements-based approach undercuts the ability of technology to function as an enabling platform for business-driven change rather than simply implementing a particular set of business behaviors. This difference is crucial because, in a business automation situation, merely implementing a set of business behaviors may deliver the benefits of standardization and reduced human labor, but it *fails to deliver on the organization's more strategic need to support innovation*. By adopting an approach that involves enabling business-driven change instead, organizations free IT to focus on delivering new core technology capabilities, while removing the overhead of IT involvement in making most business-level changes.

BUSINESS SOLUTION DESIGN

Because of the limitations of the requirements-based approach, organizations today are looking at alternatives that would allow them to both reduce the IT overhead involved in supporting business-level changes, as well as allowing business change to happen as quickly as the domain experts are capable of defining the details of that change. The key element of this new approach is the creation of a complete business-level design. This design is a comprehensive business-level artifact that captures the interrelated business elements of the automation solution that is to be delivered.

In order to deliver the kind of business design we're talking about, organizations have to develop the capability to perform what we can think of as a business-level engineering. A business-level design is more than a set of requirements; rather it is equivalent to what many industries would call a product design. Creating this level of design requires developing ways of representing the elements of a business design that are more rigorous than textual representations can ever be. These elements are things such as business processes, business rules, and business concepts, which can be used to construct rigorous workflow and taskflow designs. As in other disciplines that have followed a similar path, achieving this level of design involves defining modeling techniques that have well-defined semantics. These semantics form a framework of business design elements for domain experts that IT can preserve *literally* in the automation implementation. When these design elements are combined, we can produce a business-level engineering design or *business specification* that shows the interaction of workflows and rules, how rules reference a common business knowledge model, and how task-enablement interfaces provide windows into this work, all without reference to the underlying technology platforms.

While technology implementations have traditionally focused more on the role of technology as a labor saving device, more and more organizations are delivering services around existing products, and those services are fundamentally technology-based. Because of these demands, individuals doing business design must have a broad range of knowledge in order to be able to understand the evolving technologies that they can bring to bear on their new product ideas. Many organizations currently have a liaison-style role that assists in translating between their business and IT communities. In contrast, this business design role requires individuals who take responsibility for the complete conceptualization of the solution at the business level.

So where are organizations going to find the time and people to do this kind of design work? People often ask this question as a way of "proving" that there just aren't enough resources to follow a business design-centric approach. But thinking in terms of lots of new resources is misguided, because existing resources are already doing this work — just in an incredibly inefficient way. After all, requirements need to be completed and made consistent whether or not a business design is being produced, and this effort doesn't happen without the use of some significant resources in the IT area. The way this work typically happens is that the business design is completed in *combination* with the technical design. So what we get is one combined deliverable instead of two distinct deliverables. Organizations implicitly know this already. IT project managers frequently talk about how much time is taken dealing with "requirements clarification" and of course the dreaded "requirements creep," which is itself a symptom of the lack of a complete business design.

Generally what is happening is that IT resources, with assistance from their business counterparts, go about implicitly creating a business design without ever formalizing it and without any validation of this design by the domain experts. Instead the checkpoints are the final deliverable's behavior and conformance to requirements. The IT design documents are not designed to be fully validated by the business side, so the fact that the activities within a workflow (the business rules that constrain and direct these workflows and the underlying concepts that support them) don't have direct correspondence to elements in the IT design isn't usually seen as a concern. Essentially these individuals are performing something of a precursor to business design, which reveals that

organizations' existing resources are likely quite capable of creating a true business design. What is missing is an organizational understanding that the behavioral equivalent of a set of business design elements isn't sufficient. A business design is a key step in moving beyond this, because it provides a fully designed set of business elements.

Organizations can find people to support this kind of effort either on the business side (often in a business analyst role) or on the IT side. What these individuals have in common is the ability to conceptualize a solution to a business problem instead of just a series of feature requests and constraints. The business design role is rapidly being recognized as potentially *the* key role that organizations must develop if they are going to increase their ability to innovate and provide a rapid response to the marketplace, regulatory, and other demands being placed upon them.

Implications for Business Partners

Many of the issues I've been discussing are deeply engrained within organizations. Existing organizational boundaries reflect the lines that have been drawn separating areas of responsibility and accountability from each other. If organizations are going to allow business design capabilities to become a competitive advantage, they will need to reevaluate both the relationship between the business areas that contribute to and are responsible for business design and the IT areas that are responsible for building and operating the automation implementation.

On the business side, most organizations have a group of individuals who serve as intermediaries between the people with operational business responsibility and the IT organization. Within many organizations, these people are given the role of business analyst, but they go by many different names. Sometimes these individuals are located in a central organization that officially has the role of helping define the requirements for automation projects. Other times they are found within the IT organization, where they may be called something like "business systems analyst" or "requirements engineer." It is among these sorts of individuals that most organizations will find the people who will become business designers. However, it is unlikely that all of them will be able to succeed in this new role. The business designer role will require building a pool of individuals with a new, shared skill set. These people will be expected to go beyond recording and interpreting business needs and instead to *design a complete solution*. They will be responsible for taking the overall objectives and constraints that they are given and formulating a product that achieves the business goals.

Implications for IT Partners

On the IT side of the house, things are a bit more complicated. Because IT is a big tent, many different kinds of work and many different kinds of skills are lumped together in the IT organization. IT responsibilities typically include:

- Operating and maintaining the technology platform
- Evaluating, acquiring, and implementing new technology platforms
- Developing business applications

In the current requirements-based approach, this set of responsibilities makes sense, because IT is responsible for designing the applications, figuring out what technology to use to implement the

applications, and, finally, operating the applications. Within a business design–centric approach, though, the traditional boundaries of IT are problematic. This goes back to the intermingling of the IT design task with the business design task. Creating an explicit business design task removes from IT’s responsibilities some of the work that used to be performed as both application design and application maintenance. This outcome is a direct result of the preservation of the business concept within the automation solution. Because the business design produced by the business design task is a complete definition of the business application to the point that we could conceivably *execute* the design, the details of the business logic are fully within the control of those business experts. As a result, IT no longer needs to be drawn into doing the implicit business design work that it has traditionally taken on. In fact, in many cases it becomes possible to allow the business side to move to *direct maintenance* of the business logic since it mirrors what was recorded in the business design.

So what is the role of IT in the business design–centric approach? While organizations have tended to think of their IT organizations as tool makers, it’s now time to think of IT as the factory builders. What rolls out the end of a factory is a set of products and/or services, whether physical or virtual. That factory is how organizations increasingly produce some or all of their business value to their customers. The factory analogy is an apt one, because in the business design–centric approach, IT is no longer responsible for producing the business application’s design. Instead, IT becomes responsible for delivering *platforms* that are capable of executing the business designs that the business designers create. The words “executing the business design” mean something very specific. They don’t mean that it is the responsibility of some other group to transform these designs into a software “language” that represents the design. Instead, the intention is to create platforms that are capable of having the designs *directly loaded into them* and executed. This approach transforms the relationship between business and IT because the business side is now actually defining and controlling the business behavior that is executing. There is no longer a translation between what was asked for and what was built. By adopting this model, we follow an established path of separating *product design* from *manufacturing design*, an approach that is widely followed in the manufacturing sector. For example, in commercial aviation, an aircraft is made up of millions of parts. During design, the parts are identified and the way they fit together to form the system we think of as an aircraft is defined as part of that design. It would be unacceptable for the aircraft manufacturing group to decide that an equivalent behavior could be achieved by combining and eliminating certain parts. The expectation is that the design represents a contract that indicates the exact set of parts that will make up the aircraft when it is assembled, and it is the designer’s role to make sure that the aircraft as a whole performs correctly. We can see the signs of the shift toward this model with the introduction of business process management systems (BPMs) and business rules management systems (BRMSs). Each of these technologies represents a platform whose entire reason for being is to support the execution of business designs. What is needed, then, is a way of defining business-level behaviors for all the elements of a business design. Fortunately, a great deal of work has been occurring in this area. In the business process arena, the Business Process Modeling Notation (BPMN), a standard issued by the Object Management Group (OMG), provides a business-level way of capturing the executional definition of how processes behave at the workflow level. In the business rules arena, the Semantics and Vocabulary of Business Rules (SVBR), also a standard issued by the OMG, is a first step toward developing a standard way

of expressing something similar for business rules. But while some significant work has been done to develop standards that support the creation of these kinds of business designs, completing this picture will take several more years of work at best. For some organizations, waiting until all of this matures may make sense. However, for organizations that face competitive or regulatory pressures to make rapid business changes, being an early mover in the business design arena can clearly provide competitive advantage. Early movers are likely to find that their IT organization will be called upon to integrate and fill in the gaps in their business execution platform. With commercial products and standards still emerging, IT organizations will need to create solutions to provide as complete a business design execution platform as possible.

THE NEW PARTNERSHIP

Today, most organizations find that their existing IT investment is both essential to the organization and a major impediment to the organization's ability to respond to change. The IT cost structure has pushed many organizations to look at outsourcing as a way of controlling the spiraling costs associated with IT organizations. Yet organizations more and more are selling products and services that have an IT service aspect that makes the IT systems an extension of the business product. The existing model of custom-crafted solutions and its supporting requirements-based approach are at the core of the problem. Moving toward a platform-centric approach can allow business design to emerge as a standalone competitive discipline while allowing strategic technology investments to be evaluated using a model similar to those used for industrial facilities. This is roughly analogous to the "flexible manufacturing" model that the automobile industry has embraced over the last decade. The idea is to create a factory that has the flexibility to produce a variety of products that have yet to be defined without any significant modification of the existing production facility or equipment. The business design-centric approach requires a restructuring of the relationship between the business's domain experts and the IT automation groups. Using business designs as executable definitions that can be loaded into a business automation platform is a key step along this path. This approach is central to organizations moving beyond their use of technology as a tool for cost reduction and releasing the capability to use technology as a tool for competitive advantage. By separating out business design from technical design, organizations can enhance their ability to compete through business innovation as well as enabling their IT partners to focus on providing the breakthrough technology capabilities upon which new technology-based products and services are based.

ABOUT THE AUTHOR

Neal McWhorter leads the Business Architecture Practice at Enterprise Agility, a specialty consulting firm focusing on advanced approaches in business architecture and business analysis. Enterprise Agility provides senior leadership, training and mentoring to Fortune 500 organizations that are attempting to engineer a more agile approach to their business automation efforts in order to provide more direct control of business concepts, processes and rules to business analysts and individuals with operational business responsibility. Neal has been active in key industry efforts and currently chairs both the [OMG's Business Architecture](#)

Working Group as well as the Brainstorm Business Conference and recently led the OMG's BPMN 2.0 Evaluation Team effort. He is a frequent contributor of articles to publications and is currently working on a book on Business Architecture. Visit the BAMM.org for more information on Enterprise Agility's Business Change Ecosystem and Maturity Models.