



Points of View

Roger Burlton

Founder, Process Renewal Group
And
Chief Consultant BPTrends Associates

Rogerburlton@bptrends.com

Perspectives on Process Modeling

In this Column, I am going to take on a topic that has gotten me in trouble in the past-- the issue of modeling perspectives and BPM.

Prequel—Fall 2003

I will never forget the furor that arose in the fall of 2003 at the first BPM Conference held by DCI – Shared Insights at the Contemporary Resort at Disney Orlando. The first draft of the BPMN standard had just been released and I observed in my conference chair address that the goal of having one notation that everyone could use up and down the lifecycle of a process centric project was a noble cause. I wondered aloud, however, if business managers cared about having twenty one types of events in the notation with the same passion as programmers. Moreover, I asked how long we would last in the president's office if we tried to show such abstractions?

It so happened that in the next room a meeting of the Business Process Management Initiative (BPMI) with a hundred delegates was taking place in partnership with our conference. BPMI must have had spies in my room, because as soon as I finished my address, I was swarmed en masse (or so it seemed at the time) by a significant number of those working on the standard who'd heard about my sacrilegious remarks and felt an exorcism or at the very least a forcible conversion was called for. Given their reaction, I must have insulted everyone's honor and was put to task not to undermine their noble cause. I did not intend to jeopardize their work but, instead, to bring some reason to the argument of where use of the standard was most appropriate. I was reminded of Henry Ford's one size fits all statement, "You can buy any color car so long as it's black" and the hackneyed saying, "When the only tool you have is a hammer all your problems look like nails." My concern was simply that the varying needs of multiple stakeholders was being ignored for the sake of having just one standard. The fact that the group developing the standard was composed primarily of IT vendors and developers did not help.

After the conference, online discussions proliferated, and, eventually, others came to the view that something simpler than full BPMN (1.0 at the time), yet traceable to it would be needed to carry out the early analysis and design work. Full BPMN seemed more suited for IT specs. From that situation, the BPMN Core Notation naturally emerged as a viable toolkit. I am sure that it took more than my questioning to accomplish the new view. Nonetheless, I felt vindicated in my earlier challenge to the usefulness of BPMN 1.0 in serving a diverse BPM audience. But today some challenges remain in accommodating the many perspectives required for effective multi-domain business process solutions. So, I am going to try again to ignite some debate on the issue.

Spring—2009

The reason I decided to renew the debate is that while on vacation in Italy, I was reading Bruce Silver's new book *'BPMN Method and Style'*. My wife's response was something like, "We are on vacation, what do you think you are doing?" and, most pointedly, "Why don't you get a life?" Nonetheless, Bruce's fine job on pointing out many of the pros and cons of BPMN, as well as recommending some ways to deal with some of its missing features and constraints caused me

to rethink the purpose of notations in the first place and of process notations in particular.

The Map is not the Land and the Process Model is not BPM

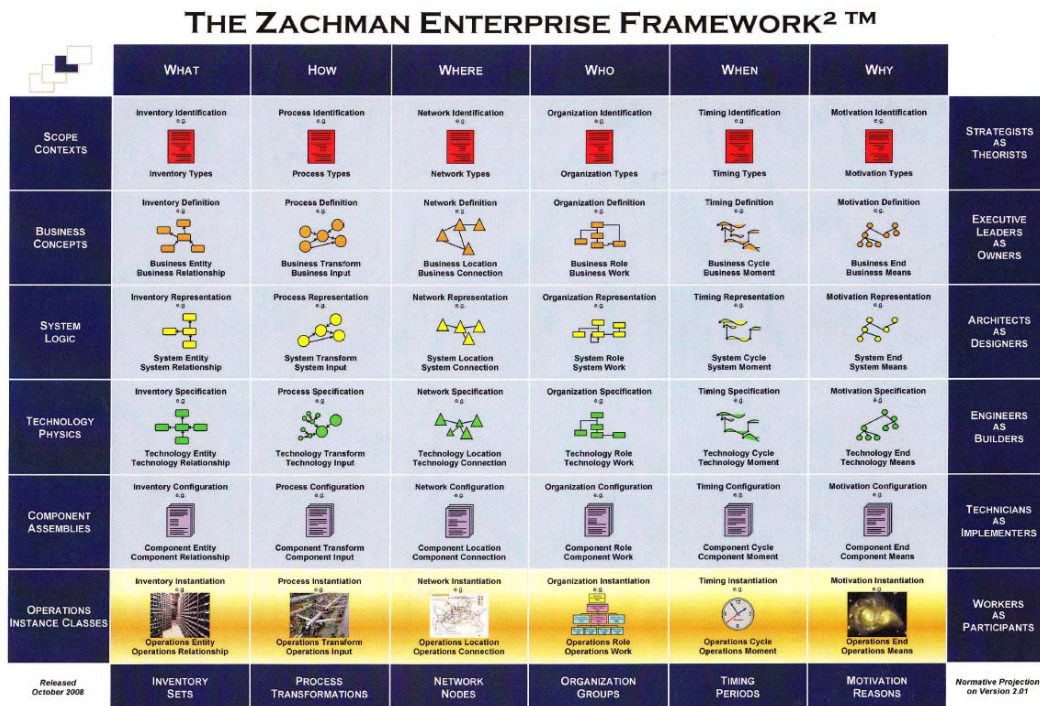
Pete Rivett, the brilliant Chief Scientist at Adaptive Inc., a metadata and enterprise architecture repository leader, has been known to say the following: “If a picture is worth a thousand words then a model holds a thousand pictures.”

This is a rather profound statement in that it suggests a process map is not in fact a complete model but just an abstraction of reality from the perspective of one viewer with a particular interest. This is easy for Enterprise Architects to accept since they are accustomed to showing the points of view of many internal stakeholders from an integrated underlying but unexposed model. But is this argument equally relevant to business process management? I would contend that it is, since BPM is more than pure process mapping. I would also contend that failure to recognize multiple BPM perspectives it is a significant reason why organizations often sub optimize process solutions or fail altogether to implement them. In a similar vein, process technology is not BPM as many of the technology vendors and some IT professionals seem to believe. Pure process maps are not the solution in and of themselves. Complete business solutions require many types of process maps and many elements working together in an integrated fashion, each with its own perspective. The flow of work across people, organizations and automation technologies, however, is an ideal rallying point or coordination vehicle for aligning or assembling the pieces needed for all aspects of the solution to have integrity and to work as a whole.

Understanding BPM using the Zachman Enterprise Framework²

A useful way of thinking about the complex multi-domain and multi-perspective nature of BPM is to consider the Zachman Enterprise Framework². This is the second generation of John Zachman’s view of how enterprises need to be architected, engineered and built for flexibility and ease of change (available from www.zachmaninternational.com). Figure 1 shows the new version

Figure 1: Zachman Enterprise Framework



of this powerful thinking model. Zachman's Model is the equivalent of the atomic table of elements that defines the set of fundamental chemical components that make up all physical things but does not deal with how they are combined into useful compounds through chemistry. Likewise enterprises and business solutions have a set of enterprise components that are combined into useful capabilities through BPM. Clearly enterprise solutions can be built to work without this model, but it is critical that the components be designed to work interchangeably with one another and assembled as needed. Otherwise, making changes becomes very time consuming and expensive.

Horizontal differences (Columns)

Each column answers a fundamentally different question about the enterprise, and we cannot build or run a functioning business without each aspect being covered. The classic six interrogatives (what, how, where, who, when, why) are the basis for this domain categorization. It mirrors the classification of information required to describe anything we build in the physical world, such as buildings and machinery. Each domain model can be articulated in terms of modern enterprises just as well. We have well-recognized this ideal in some domains as seen, for example, by our separation of data models, from process models, from location maps, from organization charts, from event charts, from strategic intent descriptions. So the question is, 'what domain models do we need to design and run a business?' The answer is of course, 'all of them'. The composition of each view into complete solutions is required, however. The challenge is that even these domains are seen differently by different parties with different responsibilities, backgrounds, points of view, interests and personal motivation.

Vertical Differences (Rows)

The description of the columns and the different domains they describe is a logical way to look at the challenge of separating the different parts of the puzzle and putting them together in innovative ways. But it is not that simple in real life. We know very well that if we build a house, there are many roles to be played, each with its own set of diagrams, needs to know, and ways of communicating to the next players who follow on. As shown by Zachman, strategists are not necessarily owners, nor are they architects, engineers, technicians or workers/ performers.

Framework Cells Alignment and Traceability

When we put together the vertical and horizontal axes we can understand that each cell has two locators on the grid: its domain (column) and its role (row).

Looking at the rows for just a moment, each also has a point of view and requires different model abstractions even in the same domain. For example, we learned long ago to separate data models at the concept (conceptual data model such as Entity-Relationship), logic (logical data model such as relational) and physics (physical such as Oracle) levels. Clearly these must have integrity as they relate to another, but the model abstractions look different because they serve a different purpose for a different audience.

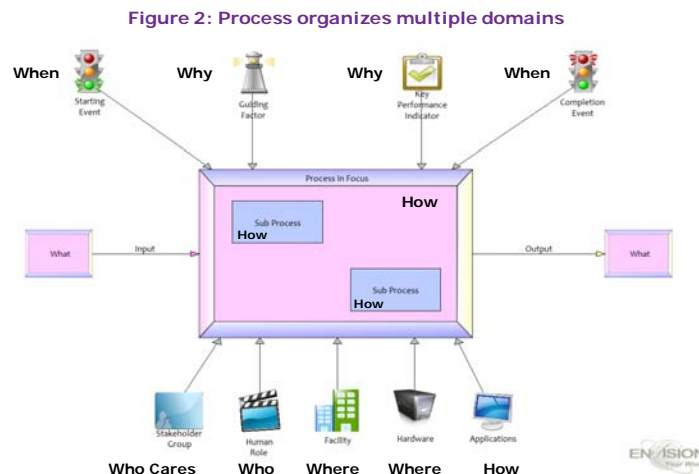
Complete composite models at any row level comprise the primitive components from each domain in the row. The components from each cell in the row are interrelated with one another in a 'where used' type of structure describing a holistic set of perspective for enterprises or complete solutions. To build anything of a complete nature, all rows must be synchronized internally, and each column must be compatible with all others in that row (my view of alignment is integrity across a row). In addition, to build anything that sees the light of day operationally, we must have all components in all rows for a column trace to one another as we transform views from top to bottom (my definition of traceability is integrity up and down the views in a column).

So what does this mean to processes and BPM?

The Implications for Process and BPM Models

Pure Process Models in the Process Domain

A pure process in column two would simply comprise input-process-output transformation, which is of course the purpose of a process. It does not matter whether or not the process is highly procedural or if it is very knowledge intensive with no clearly repeatable path articulated up front. These are all processes. In this model, there would be no location identification, no roles or organizations associated, no timing or event notification, no guiding rules and no motivation definition. It would be very simple and most likely very long lasting. It is rare to find such simple and pure process models. In fact, most views of models of processes are actually composites bringing in other related domains, principally because business processes are very good synchronizers of other things that happen in succession. They are great vehicles to assure row integrity. Figure 2 shows why that is the case.



Although it is possible to connect any domain (column) to any other one directly, process is the only one that simply or elegantly associates them all, while still staying connected to the purpose of the enterprise as well as being measurable in business terms. It can therefore be managed as a corporate asset. Hopefully, its associations with other columns will be well-formed and the interrelationships understood, thereby making it easy to change parts without disassembling the whole enterprise or business solution. Process' biggest value is as a vehicle of integrated change. We see various model types that are ostensibly process oriented, but many are not holistic when it comes to fully functional or, better yet, cross-functional change. Most models only show a sub set of all of the aspects of change that really should be on the table. That, once again, is a result of the limited perspective of the modeler who is not charged with complete business enablement but only some of the components. They have one perspective but the solution calls for many.

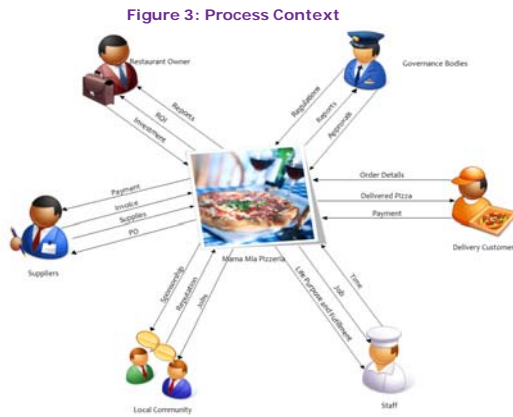
Composite Process Models

Although a pure process model as defined by column two would be very stable, it would not be very informative when it comes to designing or building composite enterprise solutions. It would be stable because it would remain true regardless of which role in whatever organization were charged with its execution no matter the location. It would be true regardless of whether it were manual or automated. It would still work (as a primitive model) despite a change in the rules applied. In a perfect world we could decide each of these changes at execution time without experiencing latency because of the latest binding of components possible, i.e., in real time. For the most part, we want to bind as late as possible for the utmost flexibility, however sometimes we have to bind at design and not at run time. The overhead cost of being flexible can outweigh the benefits. In very procedural processes, design for consistency and repeatable flow has value. In knowledge intensive customer facing processes, whose paths are not predictable or perhaps never the same among instances, it does not. Different model types are needed for each. BPMN 2.0 types of specification (some would call programming) will not work in the knowledge-intensive scenario because of the unrealistic and deterministic nature of its structure.

At this juncture, let's examine the usefulness of various process-centric primitive and composite model types by proceeding down John Zachman's rows which have clearly different perspectives. I'll discuss Context models, Concept models, Logic models, Physics models, Process Components and Process Operations. Remember other columns or domains have their own centric views of the world.

Process Context models

The purpose of models at this level is to scope the "Organization in Focus". We are essentially saying that the whole OIF is basically modeled as if it were one process or value chain. For



example, are we looking at an entire pizza chain corporation including the global head office, the establishment of franchises, the selling of shares on a stock exchange, etc., or are we only looking at a single pizza restaurant's operations. The model we may use here may be a simple context diagram showing outside relationships. For our choice of a simple single pizza restaurant a simplified one might look like Figure 3. This diagram shows the entire set of enterprise processes lumped as one box (the restaurant) with exchanges to and from the outside world stakeholders.

Process Concept Maps

Process concept maps are the business view of processes (i.e. the things we do - not how we do them). It is important to note that details gained through decomposition into finer granularity are still at this concept level, since it is viewed from a business perspective, not a technical one. Detailing does not take you down the rows; transforming to a new perspective does. For purposes of analysis, I will divide the layers into two categories: Process Architecture and Process Analysis Concept Design.

Process Architecture Layer

I prefer to keep the Process Architecture maps very simple at the beginning. In essence, at this depth, the model can appear as a visual roughly sorted list of processes that cover all the things we do. They are not organizationally aligned--merely what we do for those who care about what we do (our stakeholders). They may be categorized into types such as core (those that contribute directly to customer value creation), guiding (those that provide plans, policies, rules and controls over what we do) and enabling (those that provide physical reusable assets to repeatedly make

Figure 4: Process Architecture top level



possible what we do. This diagram type is actually a pure primitive model since it does not pull in other domains from other rows. The first step is to get the list of processes defined and agreed on before we go further. The top layer of this model is a mile wide and an inch deep; nonetheless, it covers everything that goes on in the enterprise, no matter how we are structured. In my experience, one or two layers of extra decomposition for such a notational use is sufficient, After that, we probably need a different view. Figure 4: Pizza Restaurant Process Architecture shows such a view at the top. Each of the processes represented can have its own decomposed layer of course.

Although proponents of diagrams with pools and lanes may argue that it is possible to represent the architecture with those constructs the additional information implied is not meaningful or helpful at this point. As a matter of fact showing organization or role information may actually be a distraction from the purpose of architecture since the architecture diagram often lines up poorly with the organization chart. At this point, all that is needed is a simple decomposition tree diagram just to get the inventory of processes in their proper places. Figure 5: Deliver Pizza Order Hierarchy shows an example.

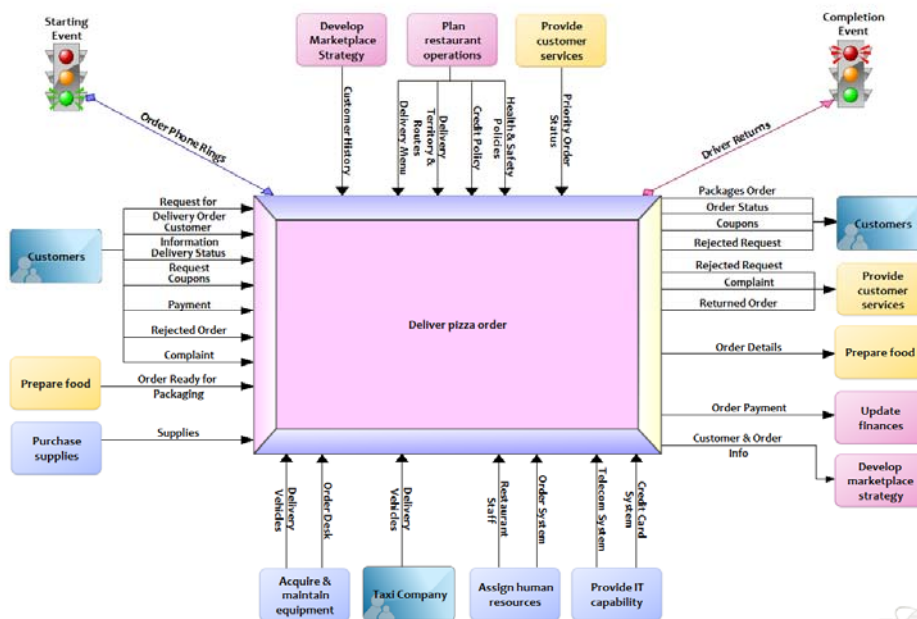
Figure 5: Process Hierarchy



Figure 5: Deliver Pizza Order Hierarchy shows an example.

Showing how each process connects to other architectural processes as well as external stakeholders is, on the other hand, often helpful. In this view, each process from the architecture chart can be shown on an individual page with the others that it sends things to (the output of its transformation) or receives things from (the inputs to be transformed or the guides and enablers that make transformation possible). The things being exchanged may be physical or informational. In essence this shows the process context for the 'Process-in-Focus'. Figure 6: Process Scope IGOE (Input, Guide, Output, Enabler) chart for the process "Deliver Pizza Order" shows an example of this.

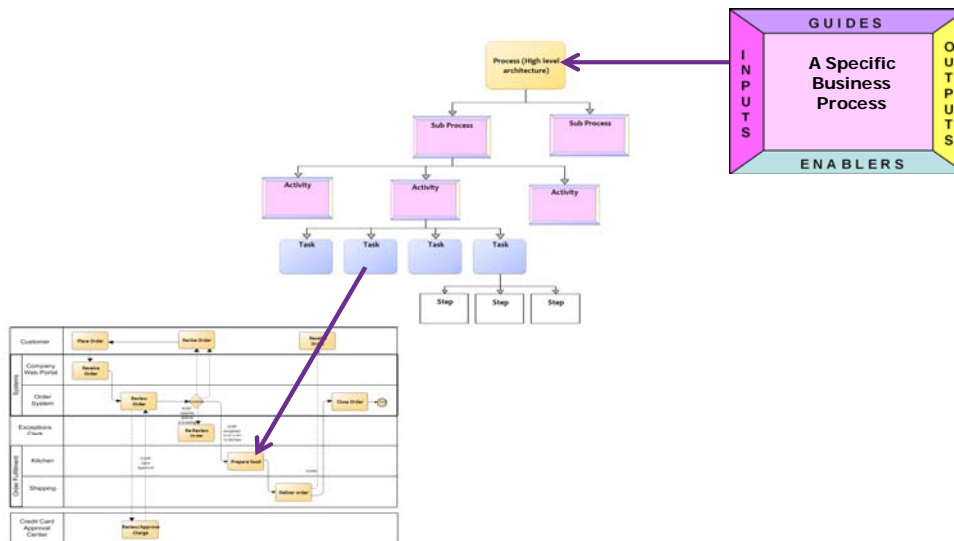
Figure 6: Process Scope IGOE



Process Analysis and Concept Design Layer

As we move to the examination of a process within the architecture, other notations become more useful in accommodating the perspectives of those doing the work at that lower layer. The Process Scope IGOE diagram is still invaluable as a starting diagram to understand the context of the process within the architecture, and IGOE charts may be useful for another layer, maybe two, to sort out the big issues within the major process activities. This is especially true when high level root cause analysis needs to examine guides and enablers as well as input- output flow for sources of breakdown. However, now we are typically trying to understand the chosen process flow more to improve its performance through new capabilities, and to do so we will need some composite views bringing in other information related to the process itself. At the bottom of the stack, however, flow-oriented diagrams are quite helpful when it is important to follow the movement of detailed work or work items. See Figure 7.

Figure 7: Hierarchy of models



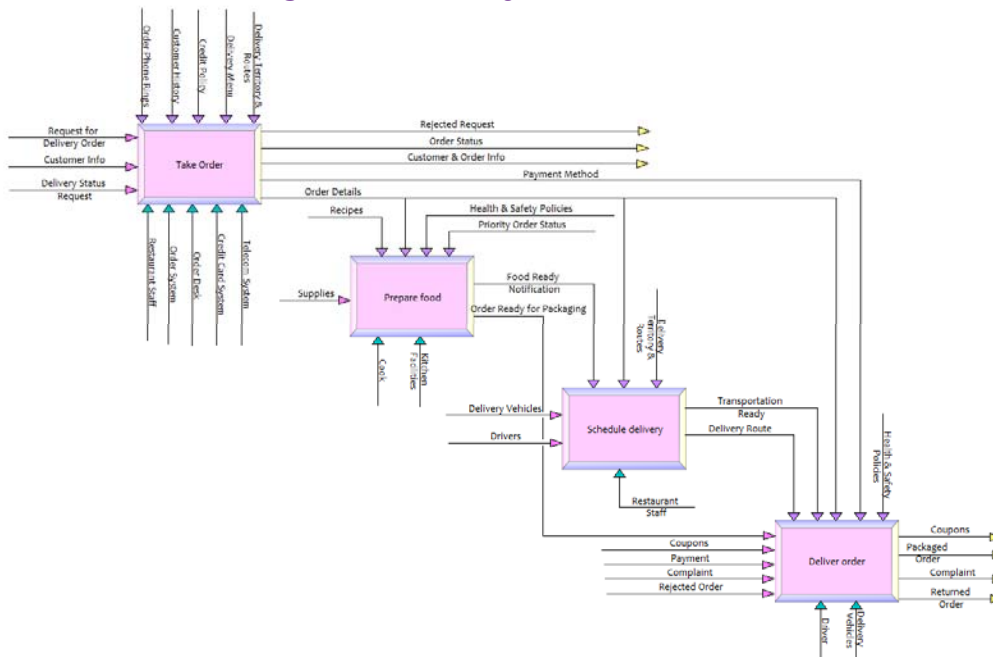
So the question is “What type of flow diagrams make most sense for the analysis and design of business processes?” IDEF0 models and Swim Lanes are two types to consider.

IDEF0

As we have moved in the past few years to the technology-centric view of business processes many people have written off IDEF0 notation. At the risk of disappointing some readers, I think this notation, originally popularized in the early eighties for manufacturing process concerns, still has a lot of value for many situations, especially for tracking information in and out of processes and also human-centric processes guided by rules. IDEF0 is clearly the foundation for the separation of the categories of interactions in IGOE charts by providing for the addition of controls and mechanisms (AKA Guides and Enablers in IGOEs) to input – output flow. The major difference is that usually IGOEs show the objects of interest as words on the shape, with each IGOE process standing alone on its page. IDEF0, however, usually shows the connection among pure process boxes with flow items on lines. Although from an underlying meta-model perspective the same objects are in play, the visual appearance is less intimidating with IGOEs than IDEF0 flows which can look confusing to business people. The IDEF0 view, however, is of great importance to an analyst. Figure 8 shows an example of how one can trace the information up and down the process activities and do an assessment on each. IDEF0 is not generally recommended as a management communication device unless you have a technically savvy audience. This class of diagram takes into consideration the other columns of the Zachman framework in a composite model across each row.

The other differentiating factor between types of diagrams with guides and enablers shown and swim lanes is that IDEF0 is very interested in data and physical item transformation, as well as

Figure 8: IDEF0 style information flow



object state changes, clearly showing what is coming in and out of a process. Swim lane approaches, on the other hand, typically focus more explicitly on sequencing and decision paths and less on the set of transformation items. The swim lanes basically answer the question, “What happens and then what happens next?” and focus less on what is transformed and the rules of transformation.

Swim Lane Diagrams

Swim lane diagrams definitely catch the attention of a lot of people in organizations since they communicate ‘the who does what’ aspects of processes. Everyone is interested in that question especially if there is the perception that anyone’s responsibilities may change. It is personal. It is another composite model that conveys important alignment and role information. These diagrams convey activity flow and dependencies but are typically weak when it comes to objects of transformation and the governing rules. In other words they do a poor job of describing what gets produced and what is needed to produce it. This is not a notation that will help the data quality advocates. These types of diagrams have been used for a long time in various guises. I was able to find industrial engineering examples from the 1930’s and even older organization and methods techniques such as linear responsibility charts which were a form of swimlane.

Rummler-Brache

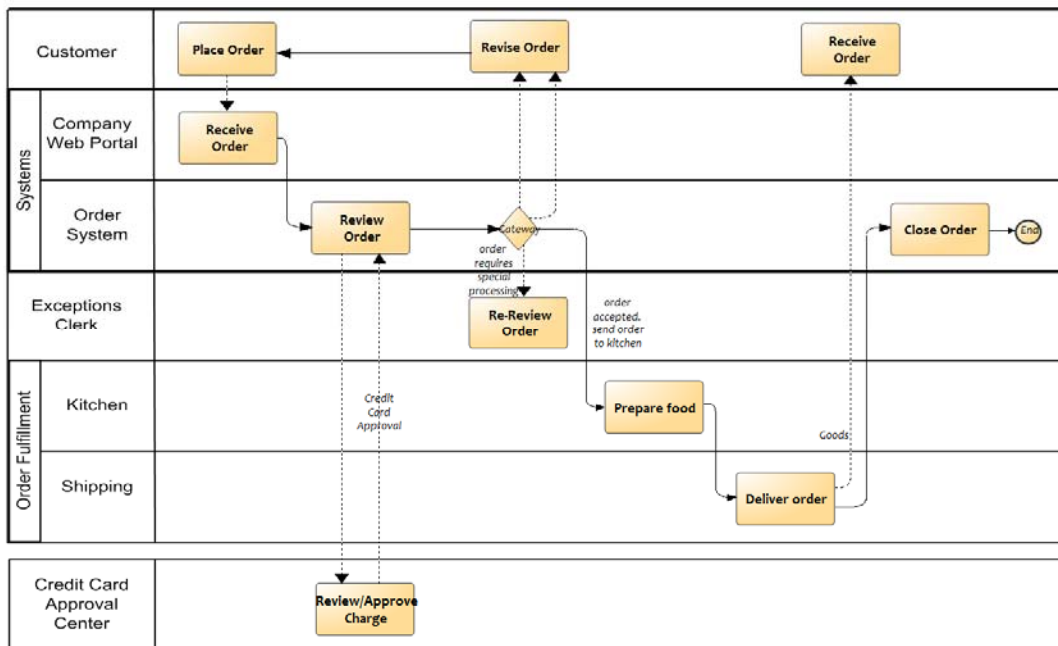
The more recent use of swim lane diagrams as we tend to know them today was introduced by Geary Rummler and Alan Brache in their landmark 1990 book *‘Improving Performance: Managing the White Space on the Organization Chart.’* One main perspective was job and organization design based on processes, hence the need for a role perspective in the process diagrams. These diagrams were mostly activity boxes and dependency / sequencing lines across lanes. They are still used extensively today by many business analysts since they are more about business and not technical mapping. Recall here that we are still working at row two: the business owners’ view.

BPMN Core

As I mentioned at the outset, BPMN falls into the class of models characterized by swim lanes but, with all due respect to the developers of the standard, it is not a business level notation in its full implementation. Those who have embraced it at the business level have discovered that most of the symbol variants, such as the multiple types of events that I mentioned at the beginning of this Column, are not relevant to the business person. Knowing where something starts and stops is sufficient. The same is true of activities and other symbols. At the business level less is more.

Alec Sharp, another BPTrends Columnist, stated in his book *Workflow Modeling* that BPMN is a powerful and rigorous notation that standardizes the drawing of swimlane diagrams but ... we [Alex et al] only use the core symbols, a small subset. ... full BPMN is best suited for drawing specification level or technical workflow models in preparation for configuring an automated workflow facility or business process management system. ... We characterize this as technical modeling versus business modeling.” I wholeheartedly agree with this depiction as do an increasing number of other highly respected leaders in BPM, Business Rules and Information Management fields. I have noticed that experienced users of both the Rummler-Brache form and the Core BPMN form exercise care in using any more of the core set than needed, relying on simple boxes and arrows and shunning diamond shaped decision gateways wherever possible since the decisions are made during an activity, not after it is completed. This convention is only useful for technology navigation. A sample swimlane diagram based on BPMN Core notation is shown in Figure 9.

Figure 9: A swimlane diagram using BPMN Core Notation



Process Logic and Physics Models

Process Logic Models are built by the designers of technology solutions that will provide business enabling capability and orchestrate the activities of the process. To do this, traditionally, designers have used portions of the Unified Modeling Language (UML) from the Object Management Group, the same body that brings us BPMN. Activity Modeling is one of UML's thirteen diagram types that will most likely be replaced by BPMN 2.0 when it is formalized.

This new version of BPMN (2.0) is even more technology focused than the current one adding significantly to the stack of notational variations for specific exception handling situations. Dr. Bruce Silver says in his new book 'BPMN Method and Style': "though most process modelers do not have a BPMS, nor any other form of centralized orchestration engine, that figment, or metaphor, is baked into BPMN." For those that have to translate a business level process concepts into technical process logic then learning and applying the full set of BPMN variants and exceptions is appropriate especially if you will have to articulate it in such a way that can be executed as BPMS choreography.

This next level of BPMN specification will do a good job of relating activity to roles, sequences and events. It will be capable of handling complex gateways and paths, but the decision logic itself will lie outside of the process map. Clearly, you will also need to have other composite models to provide a full set of solution requirements. These would include data models and rule models-- two gaping holes in the notation itself.

Although I may be over-simplistic, at this level I think of BPMN as a programming language for BPMS in the best mold of 'if-then-else' procedural logic. Make sure you ask yourself if this is what you need or want. If not, ask how you will translate business models into system logic models that can interpreted by humans and technologies for implementation and operations. Will you use a vendor's proprietary specification modeling notation, generation and incorporation of BPeL code, traditional requirements documentation and specification, or will you use prototyping / SCRUM techniques or some other form of specification. Whatever you choose, do not forget that the tools and notations at Zachman row three and four (system and technical) are not the same as at row two (business) no matter what the standards promise. Of course all levels must be traceable to one another.

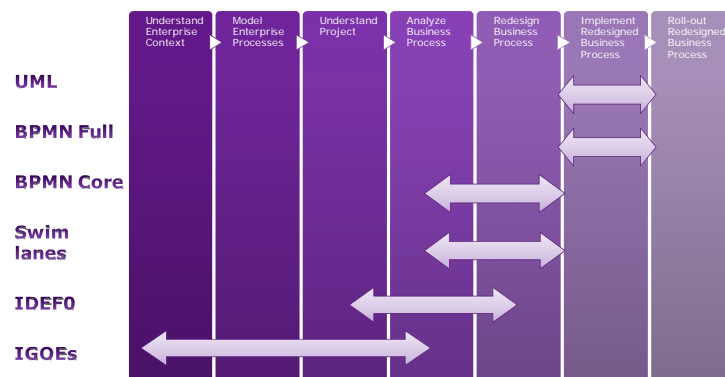
Modeling Notations and the BPTrends Methodology

In many projects performed by myself and by my associates at BPTrends and Process Renewal Group in the past, we have found that we need notations to suit the purpose of communicating with multiple stakeholders with different jobs in different domains. The true secret in this type of environment is to use simple

business process models as the heart of information gathering, communications and validation of process discovery and design. These primitive process models can then be expanded in turn to composite models that tie together various other domain requirements such as data, people, rules and technologies. Each phase of the BPM methodology has a set of stakeholders that must be able

to articulate and confirm what is being addressed. A set of traceable but different looking models will have to evolve as we make our way down the rows towards implementation. This implies that certain model perspectives, as made visible through a set of logically connected notations, are needed at appropriate places in the BPM methodology. Figure 10 shows my recommendations for using the most suitable notations across the BPTrends methodology.

Figure 10: Notational fit



Conclusion

I started out to write a simple Column, but process modeling is a complicated topic. Perspectives on multi-domain complex models are numerous. We need to show what each stakeholder at each point needs to see and no more. Remember Einstein's famous quote "A model should be as simple as it can be but no simpler". Using John Zachman's framework to think about what makes sense is useful. This means there is no one notation that will be sufficient due to the need to transform among perspectives as we progress. We need to use tools, such as Future Tech's Envision (www.future-tech.com) that was used here to show the illustrations, that has an underlying object repository to separate the metamodel data from its views as shown in the user interface. Then all professional perspectives can be presented in their full richness. Models are about integrity, ease of update and reusability. Notations are about communication. Do not confuse the two.

That's the way I see it

Roger Burlton