

BPMS 2008 – Look Back to Look Forward

Howard Smith

Just before the Christmas break, Paul Harmon pinged me, and asked whether I had a view on where BPM was heading in 2008. I promised him an article. So here we go.

In the January 5 2008 issue of New Scientist, David Peat discusses how language can limit our understanding of physics. European languages perfectly mirror the classical world of Newtonian physics. When we say “the cat chases the mouse” we are dealing with well-defined objects (nouns) which are connected via verbs. Likewise, classical physics deals with objects that are well located in space and time, which interact via forces and fields. But if the world does not work the way our language works, advances are inevitably hindered. It happened to BPM.

Let me explain.

First off – what is your definition of BPM? There have been many definitions put forward over the years. They all boil down to pretty much the same thing. First, understand how you currently work. Then, envisage a better way of working. Then, put the new process to work. Then, do it over again next year – or however often you choose to improve processes! That’s BPM.

Before I go on, I should stress (and this won’t come as any surprise to those who know my writing) that in all modern enterprises the work of people is now dependent on the work of computers. These information technology (IT) systems are rarely optional or marginal in the process. They are full participants, and without them, little of value is ever delivered to customers. Whether automating a repetitive task, enabling collaboration or to provide data analysis, computers are “in process” for the long haul. Systems are not less important than people in business processes. Systems technology cannot be ignored. Thus, any desirable change to the way people must work had better be accompanied by an associated change to the IT systems in support of that work. As any CIO knows, if human work and machine work are out of step you have an accident waiting to happen.

If a computer system imposes a rigid process on work, and it is not the *natural* process in response to the objectives and incentives handed down to employees by management, that system will fall into disuse. It happens all the time. If the users are forced to use a system they don’t like, the work will degrade, fragment, become inefficient, and, ultimately, the organization will be dysfunctional – losing customers and revenue – with rising costs. Everyone knows this. It is amazing how many people choose to ignore it. No amount of coercion will move people to use a severely ill-fitted computer system – for to do so will reflect badly on them. They simply won’t get their job done. They may use a bad system for a short while if a strong “policy” is introduced – but the policy won’t stick.

In the best run companies there is a productive and symbiotic relationship between people and systems, each helping the other along. Keeping the two *aligned* is an art. It is a major focus of most serious chief information officers. Whenever there is a need to change the “people” part of the process, one almost always has to change the “systems” part of the process. But here’s the rub:

Without BPM tools, the IT department cannot keep up. Now, with BPM tools, they can.

What has technology to do with reengineering?

During the 90s, new system implementation projects, as a result of business process re-engineering, did, indeed, require a major undertaking – not something to be sprung on the CIO as an afterthought. For to do so could delay the full benefit of reengineering for years – in some cases, half a decade! Let's give this lack of alignment a name. Let's call it

Process Design to Production Time Delay and Resource Cost

This metric can be measured. It is the aggregate man months (non mythical) required for the computer systems processes to “catch up” with the business processes. It is a real cost in at least two respects:

1. The cost of the systems change. Technician manpower.
2. The cost of not enjoying the fruits of reengineering while waiting for the systems change. That must surely be a BIG impact on the business; otherwise why is the change being considered? Businesses that are prepared to wait for reengineering must be studying the wrong processes.

As a result of stories of failing reengineering, as a result of brittle IT systems, something had to be done. It started with work flow systems. The work flow community showed the way forward. Systems began to reflect the real work. But more was needed.

The IT community learns the P word

It was the year 2000. A few of us asked the question: how could that gap be closed - forever. At the time people were calling it the Business-IT Divide. We asked: how can we arrange it so that IT systems can change “at will” (or at least – in commerce time) to reflect newly and sorely needed new process design?

How indeed?

There were a lot of suggestions floating around at the time. The IT industry was ripe for a *process innovation*.

The few of us that cared about this problem could have suggested a fancy faster way to write computer software – automation for software developers. We did not. We remembered the problems with CASE tools. And we worried about generating more and more business software, developed faster and faster. Rather, we felt the world needed less software, but software that could adapt and support more and more business processes. Ideally, we wanted a single system to support any needed process.

We could have suggested some kind of accelerated software engineering methodology – for example, taking models at one level and translating them to lower level descriptions and finally to code. We did not. We felt that would perpetuate a divide that need not exist, even if the method was “accelerated.” We worried about what happens after development, when the process changed again.

Each of these conventional options would not cut the mustard. Technicians and systems, we felt, no matter how good the software and engineering tools, would still end up being a bottleneck. Our reasons were:

1. Technicians are scarce. And they are rarely around when you need them (i.e. technicians increase “Process Design to Production Time Cost”).
2. Technicians are expensive. We sought to reduce the cost of process change. And we counted delay AND resource as part of that cost of change (i.e. technicians increase “Process Design to Production Resource Cost”).

3. Technicians don't report to the Business; they report to the CIO – and he or she controls an IT queue – and your change project might just be at the back! (= Log Jam, Arguments, Politics) Business should, in an ideal world, not be too dependent on technicians. We felt that if we went down the path of enhancing the software development process, the business side of the house could become more, and not less, dependent on technicians. And why would anyone want that (other than the technicians)?

So instead, we suggested a radical alternative.

Enter stage right – A general purpose BPM tool set.

We argued: Why not place control of process automation into the hands of the business *generalist*, and take it away from the IT *specialist*? Why not give process control to process owners and those employees that work directly for customers, i.e., to those who work with, and within, important business processes? It made sense. After all, it is they, not the IT specialists, that have the most to lose if the computer systems they are required to use (for they have no other) are unfit for purpose.¹

We also noted another benefit of taking this approach: There are far more business users than there are technicians, so that, if tools are needed, and indeed they are for any complex enterprise, process tools should be in their hands, not the programmers. Why create additional hand-offs, tasks and meetings for already-busy business people that require them to describe the processes they need to have developed to a technical specialist. They speak different languages, inevitably leading to all kinds of misunderstandings. It made no sense.

Looking for a solution we took inspiration from tools we knew worked well: data management and spreadsheet software tools. These tools are so familiar today that we hardly think about them at all. Yet they contain the clue to the solution for processes.

We do not develop new database software for each new database. We do not develop new spreadsheet software for each new spreadsheet. We simply model the data and formulae we need, press the button, and, "Hey presto", the tools adapt to us! (The programmers write the tools, not the 'use case'.)

In such *design driven* software there is a clean separation between *form* and *function*. The form is the data management software. It stays the same. The function is data management. It changes. We are thus able to perform many data tasks and we do not, as users, have to wait for a technician to help us. Same with the spreadsheet: one software package – many spreadsheets. So we asked: Why can't it be that way for all automated business processes?

It turned out it could.

But before I tell the story, it is interesting to note that before spreadsheets and before relational databases, there really was new software that had to be written for each calculation and for each data analysis that we needed to perform. Literally, we called in the Cobol programmer! As expected – delays ran to weeks and months, even for small projects.

So what changed? Why do we not have this pain today?

¹ Such is the hatred of rigid internal systems, that in some firms, employees are picking their own applications, and configuring them online. A good example is Salesforce.com and many of the applications available to extend it via AppExchange. There are hundreds of other examples. With Web 2.0, the "mash ups", this trend will accelerate. Already, with outsourcing, the idea of internally developed applications is fading fast. Now, with "Software as a Service" (SaaS) the right-sourcing trend will continue apace. Hence, a lot of BPM vendors are involved with the creation of Web 2.0 services – for it makes sense to embed process engines at the heart of the Web – if business is to get the global processes it needs. I will write on this in the future. Those interested may visit my blog at <http://saassightings.blogspot.com>

Innovation happened. The industry recognized a pattern. The pattern was that business people either needed to repeat a calculation and apply it to multiple data sets, or needed to change the calculation and try it out on the same data set. In other words, users could not predict their requirements in advance. They needed a general purpose and adaptable tool – not a fixed encoding of their current processing need. And the market – as it always does – responded with something new. As a result, that which was specific became generic. Data management and spreadsheet software were born.

While not immediately obvious, it can be that way for business processes too. The Age of Packaged Software Process is dying.

But before I tell the story, I would like to try to explain a little more about exactly what a business process is, from the point of view of a computer system.

What is a business process?

A business process is not what it seems. It is NOT just a high level description of the steps in work. It contains details, LOTS of details.

Every business process means making calculations. Every business process also means manipulating business data. For those elements we have good spreadsheets and databases – but they don't work as PROCESSES. Making the same functionality available, as PROCESSES, was part of the challenge. We not only wanted to have that functionality in our processes, we wanted to reuse the same software. And that's what, in BPM circles, is known as process projection – reusing processes from an existing system or tool as part of a new process, even to the point of adapting it to fit the new need. Sounds like magic. It can be.

In a process, data *moves* – from system to system, from person to person (via a system and network) and from person to system (via a user interface). As data moves, the process enfolds. Indeed, the process is the data moving, the act of being manipulated by many participants. We do this via computer terminals. It matters not where we are. We work on the network, on the Web, with data, accessed via Web services. (Everything else is about direct human interaction (social) – and that is not the subject of BPM.)

In traditional IT systems design – the way data moves around, from system to system, from person to person, is cast in stone, set in the logic of the software. But that has changed and is changing faster and faster. Systems are becoming configurable, by the user, for the business.

Example:

Think about email. The CIO runs an email service. The users determine which emails are sent and how email is used to get the job done. If the users were not in control, and were forced to go back to the CIO for every new pattern of email exchange, I don't think email would be all that popular. Email frees people to communicate. As with email, the same can be achieved for any business process.

BPM technology frees people to create and deploy processes. Small processes, big processes, ad-hoc processes, strategic processes, and every variant in between – it matters not. The Web inside our company or at global scale is becoming one big process engine, and we are in control.

Just go look at the web sites of the process vendors. The case studies are there. One system, many uses: that's the power of BPM – a standard IT infrastructure, supporting any number of processes, purposely built for your firm, by you.

So why has it taken so long to achieve? Partly, it is a mind set issue. People think of software packages as being the REASON to buy a process. Not so. Consider buying a BPM system, and roll your own. No longer is it a case of "New process needed?" -> "New package required."

The development of the BPM alternative to packaged software proved complex. Yet the same was true for calculations and data two decades ago.

Today the majority of everyday business calculations are performed in simple rows and columns. That's easy. But it still took a great invention, VisiCalc, to bring it to market. Until someone smart coded the finite state logic behind VisiCalc, everyone had to write individual programs for each and every calculation they needed to perform.

Same for data management: While a lot of data can still look a mess, organizing it according to relational principles allows a general purpose data query language to add to it, extend it, change its form, and manipulate it for all manner of business purposes. Until someone smart came along and coded the relational model of data management, everyone had to develop different kinds of database schema and file formats for different types of problems. It took a great invention by Ted Codd² to eradicate that divide.

And just as general purpose calculating software and general purpose data management software were tough to invent, so too was general purpose business process software.

Why so hard for process?

In any process, not only are calculations being performed and data being manipulated, but it is happening in many places, in different systems, with different users, all at the same time. Think of what you do in spreadsheets and with databases and then string out a process across the Web of your enterprise, and have many people involved. The flow is moving in patterns, such as the pattern to keep a supply chain moving. The computation, and the data manipulated, is *distributed* among many participants. It has to be *coordinated* among those able to perform the work. This distributed coordination problem was the core of the problem of inventing new BPM tools.

The complexity of this thing we call “process” is frightening. Think about this:

1. When we make a spreadsheet we only have to specify the rows and the columns and the formula we need, and we do it locally, in one place at one time. When we design a process, we have to specify who can do what, where, and when. It's a collaborative solution.
2. When we run a spreadsheet, we just hit “Go,” and it all happens right there on our PC in front of our eyes. The answer appears. When we design a process, hitting “Go” (process execution) requires massive coordination of computing resources across all of the systems involved.

No wonder then that, prior to the invention of the BPMS (Business Process Management System) the IT industry encoded business processes very carefully in packaged software systems so that only the expected process could occur. The IT industry, even now, is littered with hard-coded packaged software for every conceivable business process. Their developers do not want users tinkering with them. The programmers do not want things to break. Put another way – in these pre-BPM business systems, the programmers could find only limited ways to give users control of the process. They typically gave them configuration and option tables. But the process could only be changed within limits. And as it was changed, and deployed, it tended to become cast in stone.

This has changed, and the change will accelerate.

² A British computer scientist who made seminal contributions to the theory of relational databases. While working for IBM, Ted Codd created the relational model for database management. He made other valuable contributions to computer science, but the relational model, a very influential general theory of data management, remains his most memorable achievement.

A calculus of process

The first “process” innovation in IT came with work flow – and this was indeed a major step forward. Work flow software separated work allocation logic between system and system definition. It was based on a general purpose work flow engine. A single work flow model/diagram could re-configure the engine in an instant for any needed flow. But in other respects, there was still a lot of hard coding. This included:

- Complex calculations
- Structured data flow and manipulation

In addition to a healthy degree of hard coding, early work flow systems also limited the patterns of flow that the process engine could support. In reality, business worked in many other ways – ways that typical work flow engines could not support. An amusing example is that most work flow engines could not drive a typical email process.

To truly free the PROCESS, and place it fairly and squarely into the hands of users, we felt that everything had to be separated out. We called it a design-driven architecture (DDA). We used the word “process” to refer to literally anything a network of computer systems could do. And I do mean anything. We wanted a language for these machines that was complete and formalized and standardized. Without this specification, we felt, no one could build the fancy process engine we envisaged.

It needed new thinking.

The insight came – unexpectedly – in the form of a branch of computer science called the pi calculus, originally developed by Robin Milner, Joachim Parrow, and David Walker as a continuation of the body of work on the CCS (a calculus of communicating systems).

What on earth has this to do with business processes I hear you saying!

The aim of the pi calculus – think of it as a computer language – is to be able to describe literally any concurrent computation whose configuration may change during the computation – without breaking, and predictably (allowing for a simple transition from design to design) not just from state to state. It was new, very new.

While not developed specifically for BPM, and having a far greater significance in computer science and mathematics, a few of us recognized the significance of the pi calculus. But more than just a theory was needed. The industry needed an implementation. In fact, it needed multiple implementations if a market were to be created for a shiny new BPM technology. And for that, we needed everyone to buy into a new “standard.”

The rest is history. A new technology was born.

BPML.org – founded in August 2000 as a non-profit association – created BPML – the Business Process Modeling Language. It was the first formally complete concurrent distributed computer language that could execute any process. Later, BPML also penned BPMN – the Business Process Modeling Notation.

Intalio (co-founder of BPML.org with Computer Sciences Corporation) showed the world that the new concept could work in practice by creating the first BPML compliant process virtual machine. Intalio created Intalio n|3 – the first BPMS as defined by BPML. It was an enterprise class software system based on the new process technology base.

Founded in 1999 from open source roots (exolab.org) Intalio later donated key technology back to the open source movement and today is the custodian of the Open BPMS technologies. It is still the only BPMS to natively support the design-driven approach using

the BPMN and BPEL industry standards.

A trend was set in motion. BPML was later to be eclipsed by BPEL, but as many said at the time “BPML is dead. Long live BPML”

As a result of BPML, BPR (business reengineering) found fame in the most unlikely of places, the Silicon Valley venture capital community. Everyone wanted to adapt and position their technology for the new world of Business Process Management Systems (BPMS). Here are some examples:

- Process modeling vendors, whose software let business people draw pictures of business processes, wanted BPML and a BPMS in order to bring the process diagrams to life as new IT systems – thus increasing the value of their tools.
- Integration software vendors, whose software let IT departments mesh together disparate IT systems, wanted a language to show and edit the configured integration paths in a way that a business person could understand. And they wanted a process standard, so that different systems could “talk.”
- Start-ups, seeking new venture capital funding, jumped on the BPMS band-wagon as well. Linking their messages to “BPM” and “BPR” themes helped make the case that their technology would play well in the enterprise (i.e. sell lots of licenses).
- Workflow vendors also enjoyed the new marketing power of the “P” word and “BPM” moniker: Some sought to upgrade their technology for the brave new world of BPML and BPEL.

Everyone wanted an “in” on BPM. But not everyone understood why some were excited by the pi calculus.

Trapped in a worldview

The predominant language of “process” at the time was the language and theory of workflow systems. The workflow view of the world perpetuated a model of process as *inputs* and *outputs*. Something went *into* a process step, and something else came *out*. It was a *flow* based model.

To the vast majority of workflow theorists and to the vast majority of business people who turned up at BPM conferences to understand the potential of the new BPM technology, what they saw was the old technology of workflow – a flow of work – work items and documents on stacks flowing into and out of boxes on the screen – i.e. tasks passed from one step to another. In workflow, work flowed *through* the process. It was a view of the world that coincided with the document management systems in common use at the time. “Processes” were, for the vast majority of observers and practitioners, just electronic forms and documents flowing around.

This input-output view of work harked back to a mechanistic age dominated by factory and production line metaphors. It suited clerical work. To the BPML co-founders this was all wrong. They wanted to represent new forms of work, notably knowledge work. The pi calculus showed how. But we found it hard to explain to others. Many articles were written. Yet language was limiting our ability to explain the breakthrough.

Why so hard to explain?

In the January 5 2008 issue of New Scientist, David Peat discusses how language can limit our understanding of physics. European languages perfectly mirror the classical world of Newtonian physics. When we say “the cat chases the mouse” we are dealing with well-defined objects (nouns) which are connected via verbs. Likewise, classical physics deals with objects that are well located in space and time, which interact via forces and fields. But if the world does not work the way our language works, advances are inevitably hindered.

Working in the 1920s and 30s, the quantum mechanics theorist Niels Bohr pointed out that quantum effects are much more process-based, so to describe them accurately requires a process-based language rich in verbs, and in which nouns play only a secondary role. In the last year of his life, Bohr and some like-minded physicists, including David Peat, met a number of Native American elders of the Blackfoot, Micmac and Ojibwa tribes – all speakers of the Algonquian family of languages. These languages have a wide variety of verb forms, while they lack the notion of dividing the world into categories of objects, such as “fish”, “trees” or “birds”.

Take, for example, the phrase in the Montagnais language, *Hipiskapigoka iagusit*. In a 1729 dictionary, this was translated as “the magician/sorcerer sings a sick man”. According to Alan Ford, an expert in the Algonquian languages at the University of Montreal, Canada, this deeply distorts the nature of the thinking processes of the Montagnais people, for the translator had tried to transform a verb-based concept into a European language dominated by nouns and object categories. Rather than their being a medicine person who is doing something to a sick patient, there is an activity of singing, a process. In this world view, songs are alive, singing is going on, and within the process is a medicine person and a sick man.

This story is the closest analogy I have found to the problem many faced when trying to understand the new view of “process” put forward by BPML and its pi calculus underpinning. The language of workflow systems, dominated as it was by objects (forms, documents etc.) flowing into and out of activity steps, not only hindered the types of processes BPM systems should be able to represent, but distorted people’s understanding of a new way to represent processes. Yet those that actually picked up BPML tools found it easy ... for the tool removed the source of complexity.

We tried to explain this in the following works:

- Chapter 2 “A Walk over the hill” in my book *Business Process Management: The Third Wave*
- Appendix C in the same book
- The article “Do you GROK process?” published by BPTrends
- The paper “Workflow is just a pi process” also published by BPTrends

Others were working on similar ideas. They too found it hard to explain.

Process theorist Martyn Ould, in his book *Business Process Management: A Rigorous Approach*, went to considerable trouble to shift the perception of process away from a clerical view of administrative “input-output” processes and towards the concept of Role Based Modeling.

Ould’s ideas, developed quite independently of BPML, were closely related to and aligned with those of BPML. In BPML, the term “participant” was virtually identical to that of “Role” in Ould’s formalism.

Gradually, people began to see it

The change of viewpoint that came with BPML, Role Theory and its pi calculus foundations, led to some serious developments. Integration technology improved. Application servers improved. Process design tools improved. They all became more integrated. Workflow, however, hardly changed at all. No workflow company implemented a BPMS, even though some marketed their product as BPM.

Why?

For the majority of the integration vendors, BPML had very few side effects on their technology. The integration software was already built to run all the processes it needed to, and BPML could

easily describe them. The same was true for the process drawing tools. It was a relatively simple task to export the diagram as BPML – with some interpretation. (Note, this is VERY different from a BPML modeling tool – which can draw many processes that no typical process modeling tool can)

Work flow vendors, by contrast, were already in the “process execution” business. But they had implemented, for over a decade, a radically different model of execution as I have explained above. BPML turns up, and they need to do something very different. It turned out to be a major software development undertaking for them. Few made the attempt. And this is why the majority of work flow vendors chose to adopt BPML mostly for marketing and only in a minimal technical sense. They typically used BPML or BPEL to expose the interface to their work flow engine so that it could participate more fully in a complete business process orchestrated, typically, by an integration engine.

Thus began some industry consolidation.

The most famous workflow vendor, Staffware, chair of the WfMC (the Work flow Management Coalition which set work flow standards), was acquired by integration server specialist, TIBCO. FileNet, another significant player, was acquired by application server specialist, IBM, to boost their content management offerings.

To this day, no workflow company operating at the time of BPML founding (2001) has developed a full implementation of BPML or BPEL. The task is simply massive. It turned out that the IT infrastructure and application server companies, such as IBM and Oracle, had more pieces of the puzzle. This should be of no surprise, since BPMS and open source specialist, Intalio, placed a great emphasis in their solutions on transaction management in the distributed computing environment and, later, allowed the BPMS to coexist with both open source, and commercial application server stacks.

Ironically therefore, the BPMS is, today, more a feature of IT infrastructure products, than of work flow products. And this, by the way, is also why SAP has taken such an interest in BPM – despite making little progress on an implementation itself.

SAP, whose business was predicated on pre-packaged software for common business processes and which is now predicated on component software that is assembled to make a complete solution (process), saw early potential in the BPMS concepts. The reasons included:

1. SAP was frequently criticized for the long lead time of SAP projects. It was also criticized for the difficulty of adapting SAP processes to user patterns of work, and of changing the systems if work patterns changed. This led to adoption and maintenance problems for new SAP-based systems. The company was therefore already working on ideas to make its software more configurable. Along comes BPMS. It was a useful marriage.
2. SAP was a bespoke and proprietary world unto itself. With the growth of open systems, Web standards and the Java platform, SAP needed to substantially overhaul its development platform. BPMS was an important piece of the puzzle. It was built to standards in every aspect.
3. SAP was facing new competition at the level of the platform. More and more processes were being built on integration or application servers, not SAP. SAP responded with Netweaver – an integration platform for SAP and the basis of claims for a coming BPMS. SAP therefore moved to embrace the concept – albeit using its own marketing terminology.

SAP is embracing the BPMS, but slowly. With developments beyond Netweaver, and new products yet to be announced, code name Galaxy, SAP has the potential to take BPM into the mainstream in a larger number of enterprises. According to insiders, it is “amazing to see all

these [SAP] processes being modeled in ARIS and the whole thing taking on a much more process and model driven approach, rather than weird configuration rules”

Galaxy – a BPM with BPMN modeling, etc., includes browsing of services registry and a cute interface – just like the old days of BPML. Hopefully it will go main stream.

So what’s ahead for 2008?

There is not a single serious corporate enterprise architect today not conceiving or building a design-driven process based architecture. Many look to learn via Intalio and the open BPMS movement. The moves by insider gorillas have confirmed this is the winning approach.

SAP’s interest in BPM is triggering similar moves by other packaged software firms. Those who previously built code logic for specific business processes have started separating it out, implementing or embedding a generic process engine (rather than trying to build one). They are adding fancy UIs to allow users to tinker with process.

Even some legacy systems, in Fortune 100 firms, are being overhauled. Monolithic code is being replaced by either a workflow engine, a rules engine, or a business process engine. Rather than the system being the process (set in stone in software), the system is the engine of any process (able to adapt under business control).

At the same time, a new breed of BPM products, pulled together from workflow, rules and integration engines, are creating BPM suites, products that are not BPMS, but good enough and highly usable and functional. They are letting business process create many new processes in a fraction of the time it would take to acquire a new packaged solution. Literally – companies are creating processes in less time than it takes to research the market for a package, make the business case, place the order and only then to find out it is not quite what they needed.³

BPMS or BPM suite – it’s a matter of horses for course.

The trend is set to continue – for a long time.

Mature full-featured data management (RDBMS) products took twenty years to mature. Why should BPMS be any different?

A long cut – via Web services

Despite the existence of the BPMS, the vast majority of the world’s developers are still building processes piecemeal, exposing little pieces of process here and there, and then finding ways to connect them. It called Web services, and it reminds me of the days when we stitched together files to give the illusion of a database. But whether we like it or not, Web services are here to stay. It’s the new programming. And Web services are the flip side to a process language.

Think of a soda drinks machine. The coin slot and the collection tray are the Web services of the process that let you satisfy your thirst. Get it? You made the connection between process and service when you reached out for a drink. Think “Lego brick” software. You were the BPMS. You could have walked away. You were managing the business process. There was no BPMS. And that’s the case with Web services today. Most are just that, Services, and nothing is driving them.

Yet Web services derailed BPML.

Web services consist of a complex set of standards for exposing and linking pieces of a process on a network. The Web services trend – a far bigger trend than BPMS - drew great interest from the platform and application server vendors. IBM and Microsoft knew they had to control the Web

³ A set of case studies was included in the paper “From CIO to CPO via BPM” published by CSC.
<http://howardsmith.editme.com/bpm>

services standards. They took a proactive interest in everything to do with Web services, and penned several important standards. As they reached towards BPMS, via Web services, they hit BPML head on. Unfortunately, a marriage did not occur, and the gorillas felt compelled to develop a competing language to BPML. That language is called BPEL – the Business Process Execution Language. How close is that!

BPEL was promoted using marketing dollars and eventually eclipsed BPML. Some of us were upset.

But why all the fuss about the details of technology? What has any of this to do with business process? Back to the story.

In the beginning

...there really were no computers (zero, zilch), so companies could improve processes without any consideration of computers. We called it the “First Wave” of BPM.

Then computers came along, and helped us to automate all kinds of business tasks. In this era – a “Second Wave” of BPM – it became important to change the computer systems as soon as possible after the business changed. This was the 1990s, the era of ERP, and numerous packaged solutions were “merely” participants in the process. All was well if their part of the overall process did not have to change. But, as automation became more and more important to the CEO (cost, speed, efficiency, etc.) it was inevitable that pretty much any business change would imply a deeper systems change.

By the end of the 90s the tail was now wagging the dog. Much to the chagrin of BPR diehards, systems automation and IT development were now driving the business, not the other way around. Something had to be done to make systems more agile, more amenable to change under business control.

BPMS is one way to do it.

In itself, a computer able to change readily to changing business circumstances is not new. Long before BPML came on the scene, people were making computer systems easier to use and adapt. To be clear – not all user adaptable software requires BPML.

Some of you will remember the early 80s. Remember when the sales director first understood he could get his sales figures ahead of the official channel by running his own queries on the database on the mid range server. This trend toward empowered users continues to this day. Super users are replacing programmers. But that does not mean that BPML is not needed, as we shall later see.

And so a super user emerged ...

As reported in the December 2007 issue of *Computing*, “Programmers are becoming a rarity in many IT departments, and in their place is a new breed of super user who is able, with the right tools, to build software quickly in response to business need.” The growth of application-building programs, with graphical, task-oriented user interface, reusable templates, and automated design assistants is revolutionizing the way businesses can approach the development of new IT systems. “Business people can [now] create software matching their needs as soon as they emerge, avoiding the transaction costs and delays associated with having other parties in the loop.”

Sound familiar?

In the article, Arista Insurance, a mid sized firm, is quoted as saying, “User programmers can be a positive differentiator in the struggle for speed to market. With the right tools, our people have the insurance knowledge needed to bring products to market, which programmers do not have.

It's about a combination of skills: We have business analysts who have a background in underwriting or have worked in insurance distribution, but also have a flair for programming."

Is this BPMS?

Not quite.

When BPM was first proposed, a strong objection came from those practitioners and consultants who had past experience of BPR (Michael Hammer's reengineering). They argued, incorrectly, that business process change had little, if nothing, to do with computers. While the same people acknowledged that IT systems were important, they preferred to ignore them during their reengineering deliberations. They were often seen to throw requirements for new or changed IT systems over the wall to the CIO, once the business had worked out how it wanted to work. What they missed, was the potential for a "Third Wave" of BPM.

In the "Third Wave" – the system is not just a participant, but is also the tool by which the process is understood and changed. This is hard to understand. Let's state it another way:

Take #2

If the computer is limited to automating its little piece of the process, and no other, then as long as it embodies tools for super users to change it, it can be kept in line with the business practices. The business changes, and, as quick as a flash, along comes a super user (or a programmer with fancy accelerated development tools), and brings the system back to where the business needs it to be. But, such a system cannot measure the effectiveness of the end-to-end process of which it is a part. It is blind to the scope of process beyond its own code. It knows its own piece of the process, and it knows how that interacts with its users.

What about the rest of the process?

It has been estimated that 95% of what happens in a typical business process can never be automated. And this is why and where the BPR practitioners are right. When they say "I try not to get involved in IT" they are being entirely sensible. The vast majority of what a business process is *is* the way people do things. The computer systems are, in reality, minor players, even if they are stumbling blocks to change. So this is why the BPR folks also get it wrong. At every BPM event, I'd hear someone say, "It's not the technology, stupid." Unfortunately (for them), it is.

Even if the bulk of a typical business process is non-routine and non-automated, that does not mean it does not need IT support. Nor does it mean the systems in use won't trip it up when change is needed. Flip chart reengineering is not enough. But just moving the business analysts to an electronic drawing tool is not enough either. The process lifecycle (covering automated and non-automated processes) must be modeled, and must also be able to be executed.

Do not confuse the word "executed" with "automation." Process Automation merely repeats a process. Execution means more – for a digital description of a process (remember – 95% of which may have nothing to do with IT automation) can, nevertheless, be brought to life in the computer.

Cars are not computers, yet they are modeled, simulated, and put into automated manufacture, using computers. The same is becoming so for business processes.

There are many reasons to digitize executable business processes:

1. Visualization. Computers are great tools for helping understand what is happening in business. We don't just want static pictures of the process – we need live data.
2. Analysis. Computers are great tools for analyzing the performance and design characteristics of business processes. We need simulation. Without this insight, our reengineering decisions may be flawed.

3. Design. Since the boundary between what is human practice and what is system automation shifts over time, it makes sense to design for both in *one* environment, and make decisions about deployment of the process roles once the design is clear.
4. Measurement: Only if the whole process is modeled can we measure whether the results in practice are meeting our performance indicators. There is little point measuring only the reliable automation of computers! We need computers to watch over the unreliable steps of humans.
5. Knowledge: Process is the new data. Knowledge workers live with, work in, and manipulate business processes. If they are denied digital process tools, what shall we give them ... pen and paper?!

There is one last reason why we need the whole process – that which can be automated and that which cannot be automated – to nevertheless be digitized in our computer systems. That reason is change.

It is not so easy to change a business. Top down diktats, incentives, change programs – we all know how difficult it can be to get people to do what you want and change the way they work. And this is why, in company after company, processes are moving online. Once online, they become the way people work, even if they only strictly “automate” a part of the whole process. Through such digitization, companies have the levers they need to foster and accelerate change in their organizations.

The “Third Wave” of BPM is a new way to automate, but it’s not only that. It is also a new mode of control over the process, comprising eight broad process capabilities: discovery, design, deployment, execution, interaction, control, optimization, and analysis (forever in a loop of improvement and re-design). The hallmark of the BPMS, therefore, is the combination of automation and design in one package. Why would anyone want it otherwise!

“As workers and as consumers, both online and offline, each of us is enveloped by myriad business processes – the intricate, dynamic, ever-changing manifestations of the economic activity of companies. Whether we are disinterested, or actively engaged, in these processes, in large part determines the wealth of those who weave them. Companies are looking for secrets, skills, and tools that will enable them to create and mesh together business processes that are so outstanding that customers will pay to see them, time and time again.”

I first wrote those words in 2003. They were published in a book I authored with Peter Fingar, *Business Process Management: The Third Wave*. My view has not changed. What has changed is that the tools are now at hand. Super users are replacing programmers. Super users are replacing architects. Super users are replacing consultants. Anyone who is not a super user and who refuses to become one is denying the pivotal roles of information technology in process change.

The super user is the programmer.

The super user is the business architect.

The super user is the process change consultant.

And now the super user is in charge

Don’t *bridge* the business-IT divide in vain attempts to make it easier for technicians and business people to work together. That way will never work. It will not eradicate the Process Design to Production Time and Resource Costs, for those costs are governed not by the productivity of programmers (no matter what programming tools you give them) but by the productivity of business people to make change happen in their organizations.

Rather, *obliterate* the divide by placing process control into the hands of users. Then, the IT guys can continue to do their job without being drawn into the process change agenda, for the correct role of the CIO in the era of the CPO (Chief Process Officer) is to provide, to the business, a “super-user driven process design, execution, and analysis capability” (a.k.a. BPMS)

Think CAD/CAM for the P-word, and you'll get a strong idea of what lies ahead in BPM in 2008 and beyond. Here's the analogy:

CAD/CAM takes product design and moves it seamlessly to manufacture, unleashing the value of *product* innovation. BPMS takes process design and moves it seamlessly to execution, unleashing the value of *process* innovation.

Two things will happen next:

BPMS will mature. It will be the bed rock of all new systems development.

BPMS will grow in scope. Watch for new capabilities in BPM products.

And after that?

Today, the BPMS can manage the lifecycle of the process in execution. Tomorrow, the BPMS will manage the lifecycle of the process in design. Watch for tools for process design innovation.

Think systematic innovation for PROCESS and you'll get the idea.

Author

Howard Smith is the author of two books: *Business Process Management: The Third Wave* and *IT Doesn't Matter – Business Processes Do*. He is CTO for CSC in Europe. He was a co-founder of BPML.org. If you found this article interesting, write to him at hsmith23 at csc com. He can be found on the Web at <http://howardsmith.editme.com> with links to other papers.