Workflow is just a Pi process

A breakthrough in the representation and execution of business processes inspired by the Pi Calculus, and enabled by new Business Process Management Systems (BPMS)

Howard Smith & Peter Fingar

(Available as a PDF download from http://www.bpm3.com/picalculus)

Abstract: There is much talk today about a business process management (BPM) rEvolution. The revolutionary part is about a new category of software known as the business process management system (BPMS). The evolutionary part is about using the BPMS to exploit existing business and technology assets in a way that creates new value. Along with any revolution comes confusion. What exactly is BPM? Isn't it just workflow technology, which has been in use for twenty years, plus Web services? Why don't we describe what is going on today as the "new workflow rEvolution," a subtle extension of workflow systems? To answer these questions, we explore the foundations of the workflow paradigm, and describe the paradigm shift in technology that is needed to overcome limitations of workflow systems to build and deploy robust business process management systems—the kind of information systems that businesses now demand as new sources of competitive advantage in an ever more uncertain and complex global economy.

The world of workflow

> *Severe and well acknowledged problems of Workflow Management Systems stem from their rigorous and formal nature. Implementations of workflow tend to be coercive, isolationistic, and inflexible; whereas the natural interaction of people frequently incorporates flexibility, opportunistic behavior, social awareness, and compromise. -- Skip Ellis, BPM'03, Eindhoven*

The word "workflow" is etched into our collective consciousness … the *flow* of *work*. We each have a deep-seated understanding of what this means, based on our experience at work and our work with existing workflow technologies. In our respective organisations we spend a lot of time with documents and forms. We pass documents and forms to each other in support of our daily tasks. We do this in a myriad ad-hoc patterns using electronic mail and in more prescriptive ways using workflow management systems. Workflow systems structure our document exchanges so that our work has rigor. We are enslaved by workflow and simultaneously empowered by it. Such systems let us set out the desirable flows of work and our computers help us with tasks that can be so automated, freeing us for more creative and productive activities.

Intuitively, it seems that all possible business processes can be supported using just these ideas of flow, documents, forms and routing. After all, we can enrich the automated flow of documents using all kinds of business rules. Information of interest to us can be prioritised, classified, sorted and distributed to those in key roles so as to seemingly meet every nascent business need. We can also include key back-

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

office IT systems in the flows, even machines on a production line or trucks in a logistics supply chain. Where IT systems or machines can help us in our business, we enlist them to do work for us by using workflow systems to shunt information to and fro. We use the same technique to shunt information onto and off the work *task lists* of our work flow desktop. We are driven by work, and we create work. The work *flows*. We even extend the workflow model with schemes that route and allocate work according to business strategies, based on resource capabilities, responsibilities and availability. It is an understatement that, with workflow management systems (WFMS), we can create very sophisticated flows of work. But there is a catch.

Workflow systems are not based on a single model of workflow

Today, most enterprise applications (for example, ERP systems) include a WFM component, and workflow engines have been used as the control elements at the heart of enterprise application integration products (EAI brokers). Workflow is far more than an aid to manage documents and forms routing—it has become a systems development platform in its own right and a way to develop new business applications. Advocates point to the fact that 75% of workflow projects succeed while 75% of application development projects fail. It appears that defining a business system in terms of work item flow is easier, and more flexible, than trying to develop the same functionality as bespoke software. This is not surprising, since the flow of work among people, systems and machines is a natural way to envision, design, build, manage and operate an information technology infrastructure. It is closer to the way business people think. Such technology is mature, well understood and widely deployed, although it has been more successful in some industries than in others. Why is this?

For years theorists have studied so-called *workflow patterns,* the patterns of work that occur in business and how these can be supported by WFMS. Vendors of workflow solutions strive to support as many workflow patterns as possible. The very best workflow management systems support a rich array of patterns that can be used to construct elegant system behaviours. So why is that workflow systems have not been used to develop all software? Why are we still using Java and a host of other computer languages? One reason is because different workflow systems implement workflow in different ways. The implementation of workflow, by different vendors, takes many forms.

Despite the efforts of standards groups, particularly the Workflow Management Coalition (WfMC.org), many workflow systems are as different from each other as they are from Java programming. For this reason, CIOs that deployed workflow where nevertheless unprepared to commit to the workflow model as their primary systems development methodology. To do so would require them to commit to a single workflow vendor because of differences between workflow engines. They lacked confidence that they could move workflow models between the WFMS of different vendors. This reality has limited the market adoption and applicability of

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

workflow[1]. Differences in workflow semantics also fragmented the market for workflow solutions, with many niche vendors still finding a role for themselves because of unique workflow "features" only found in their solution. The situation has been quite frustrating for some workflow theorists as well, who have suggested further standardisation of the workflow model through the publication of proposals for new unified workflow languages, such as YAWL, Yet Another Workflow Language [Ref 3][2].

But there are deeper reasons why workflow technologies cannot be used to build all possible software processes, even if the industry all agreed on one workflow model. The flow of work, whether among humans, systems or between both, is only one possible way to think about process. To understand why depends upon how you view the world of business processes. For as it turns out, workflow systems view the world in a way that limits the types of processes they can support. This is a fundamental limitation, inherent in the classical workflow model itself and is the reason that today theorists are proposing a host of extensions to the workflow model to make up for its deficiencies. Some of the most common processes we use in business cannot be modelled and deployed using workflow engines.

Recently, a new way to think about all processes, called the Pi Calculus, has emerged from theory into robust implementations—Business Process Management Systems (BPMS). The BPMS is a new category of software, as WFMS and RDBMS were before it. To understand the platform shift, we are going to describe a rather trivial process example, electronic mail. While we would not use BPMS, or WFMS, to implement electronic mail, studying the way electronic mail works is helpful in understanding new possibilities for supporting the automation of all business processes.

Workflow semantics cannot model the majority of business processes

Consider electronic mail as a process, a process upon which we all depend. With the advent of viruses, the spread of spam and the menace of the "reply to all with history" button, some may regard email as an undesirable business process. But for the purposes of this discussion let's put these concerns to one side. How does email work?

We send email to you, you pass the message to third parties and, through this exchange they are able to get back to us. How does this happen? By receiving email, or more specifically by receiving an email address, directly or indirectly, we acquire the capability of giving, to third parties, the capability to communicate with others linked to that email address. (Read that last part again, as it's important.) This is what makes email work.

---

[1] Contrast this to the success of the relational database, based on a common language, SQL, and a common data representation. The RDBMS can be argued to be the most successful category of software, unpinning the success of myriad 'data centric' applications, not least of which is ERP.

[2] The authors would be interested in hearing from anyone implementing the full semantics of YAWL in an industrial strength solution. Contact howard.smith@ontology.org

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

We give a *name,* in the form of an email address, to others, and this gives them the ability to communicate with yet other participants in the thread of the conversation—opening up the conversation so that it extends, over time, to involve new participants that contribute value to the process.

The email thread represents the history of the process. In each mailbox the thread represents the history of individuals unique interaction in the process. The email addresses represent the participants in the process. Through this simple model, which is implemented within email servers like Microsoft Exchange, a dynamic world of digital conversations becomes possible. Now, what's really noteworthy is that, without exception, even this simple process cannot be directly modelled and deployed using a classical workflow engine. The reason is that the email process exhibits so-called *mobile* behaviour.

Processes, but not workflows, exhibit mobile behavior

Mobility is a property of most, perhaps all, processes—a phenomenon first recognised by ACM Turing Award Winner, Robin Milner, who, working with colleagues David Walker and Joachim Parrow, developed a formal theory of mobile processes: the Pi Calculus. The term *mobility* refers to the way in which processes evolve as they execute, through the exchange of information among participants whose relationships evolve as a result. In the email example, the mobility of email addresses changes the links between people, determining what they know, whom they know, and how they found out.

Milner observed that the world around him, at least as it relates to the way processes are embodied in computer systems and networks, comprise separate *computational* and *communicating* elements at all levels, from the micro to the macro. For example, within a microprocessor device, the CPU computes as it communicates using registers. Putting devices together to make a board level assembly, the devices compute as they communicate via the copper tracks on the board between components. Putting these boards together to construct a computer, they compute, as they communicate via the back plane bus. Then, we deploy computer programs on the computer and they compute as they communicate, using messages or shared memory. Subsequently we develop business applications, and they compute, but need to be integrated so that business data can be shared and communicated among them. (We call this EAI today.) Finally, we place these computing systems on wide area networks and they communicate to implement electronic mail, file sharing, the World Wide Web, E-business, EDI and a myriad other business-to-business or system-to-system processes.

It certainly appears that, as far as computer science goes, the IT industry at large treats computation and communication as two very different, and very distinct, disciplines. Some devices compute, some devices communicate. When Milner and his colleagues began to think about this, they posed a very hard question: could it be that computation and communication are merely manifestations of the same thing? They sought the equivalent of a Grand Unified Theory in physics. They called the thing they were looking for a *process.* They found they could build processes with processes.

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

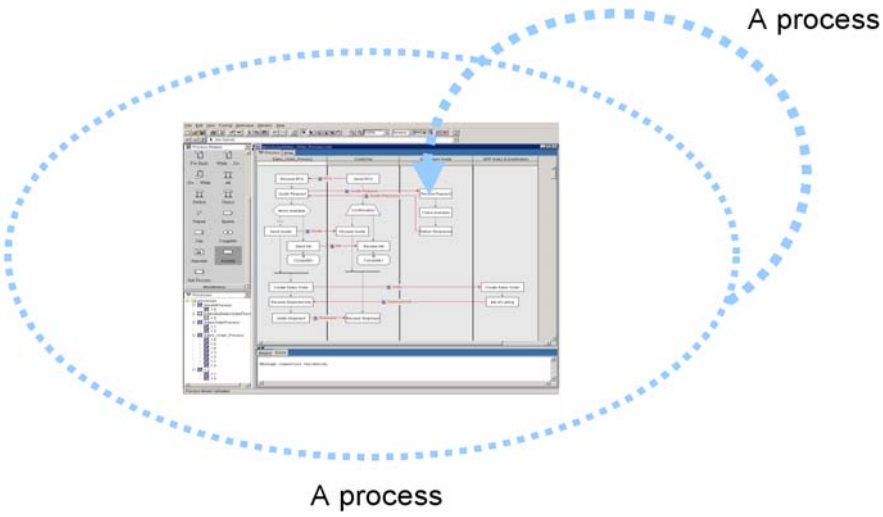

Figure 1 – A process is made of other processes.

It was these theorists' insights, built on the shoulders of previous pioneers in concurrent computation such as Gul Agha and Carl Hewitt, that ultimately led to the development of the Pi Calculus, and, many years later, the foundation for BPMS. The development of the Pi calculus itself was a multi-year effort, resulting in a formal model for all processes.

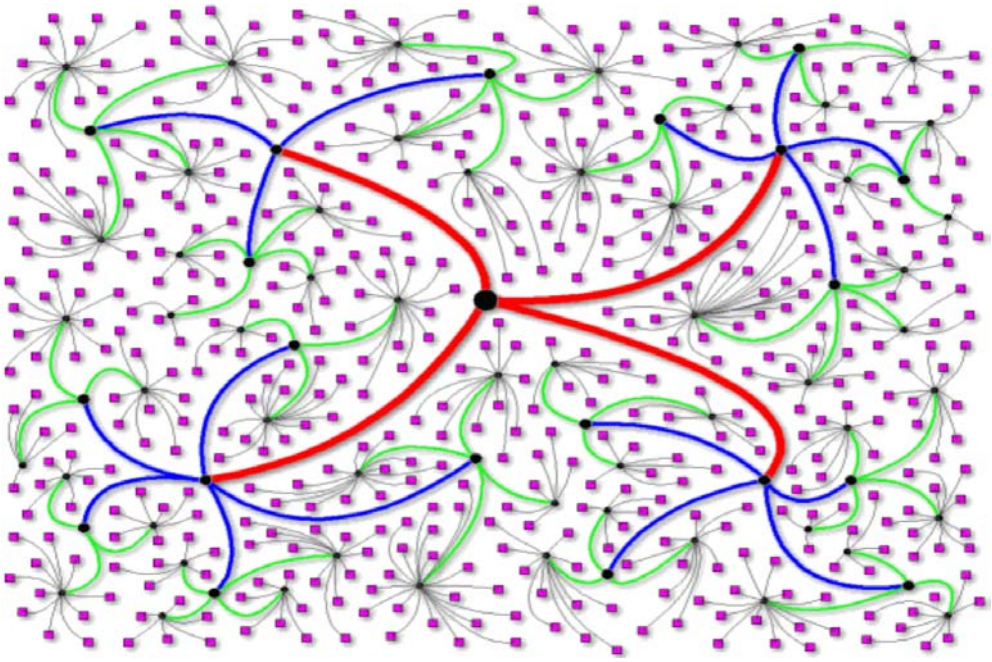

Figure 2 – Milner and colleagues observed that processes exhibit mobile behavior, dynamic links established through the interchange of information.

A new foundation for business processes

Milner was motivated by the search for a true unification of computing and communication and he gave it the name *informatics* [Ref 4]. He knew that informatics would provide a new understanding of, and options for, the implementation of concurrent distributed processes. This had been an active area of research prior to the

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

identification of the Pi Calculus, both by Milner in respect of his earlier CCS (calculus for communicating systems) and the work of others, for example, Anthony Hoare's process algebra CSP (communicating sequential processes). Before these innovations, the prevailing Lambda Calculus had been the underpinning of our understanding of single-threaded computation. The Pi Calculus now provides a general theory of *interaction* within and among multiple computational threads.

Processes, Milner observed, consisted of many elementary parallel, interacting, communicating threads. The behavior of the process, and its result in the environment, was governed only by the information passed between the elementary threads. Even when Milner looked at something that appeared, at first sight, to be single-threaded he saw instead a way to understand it using *parallel* constructs. For example, adding an item to a list of tasks—a common requirement in a computer—can be considered as an interaction between two processes, the head and tail of the list. The list grows as its separate *process participants*, the items in the list, communicate with each other, exchanging pointers. This is a very different understanding of the list's behavior than our usual notions of a list data type operated on by code to manipulate the pointers.

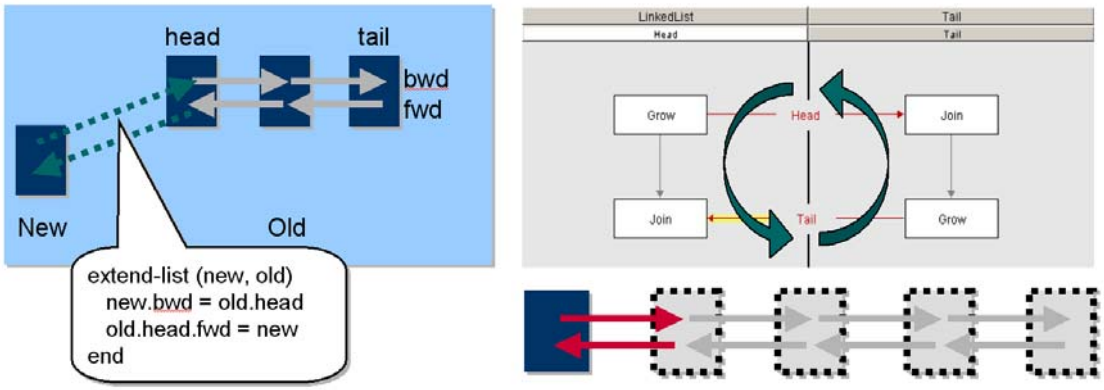

Figure 3 – Two ways of looking at a listed list. On the left, as separate data structure and software code (computation and communication). On the right, as a process, in which the list grows as its head and tail interact. Such elementary processes are said to 'proceed', i.e. execute. There is no distinction between computation and communication.

By adopting this approach to process representation, arbitrary distinctions between what is communication, and what is computation, begin to dissolve in front of our eyes. In the world of business processes, the unification is between control-flow, and data-flow, and between participants that exhibit both types of behavior. If this seems a little confusing at first, it's necessary to understand that in the world of Pi Calculus, all participants in a process are themselves processes. This is true whether the participant is an 'unlively' entity such as the integer "1," or as dynamic as the behavior of a complex, end-to-end business process such as order-to-cash. Thus integers, stings, objects, workflows, procedures or indeed any other digital or computational entity or service can be considered to be an abstract data type called a *process*.

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

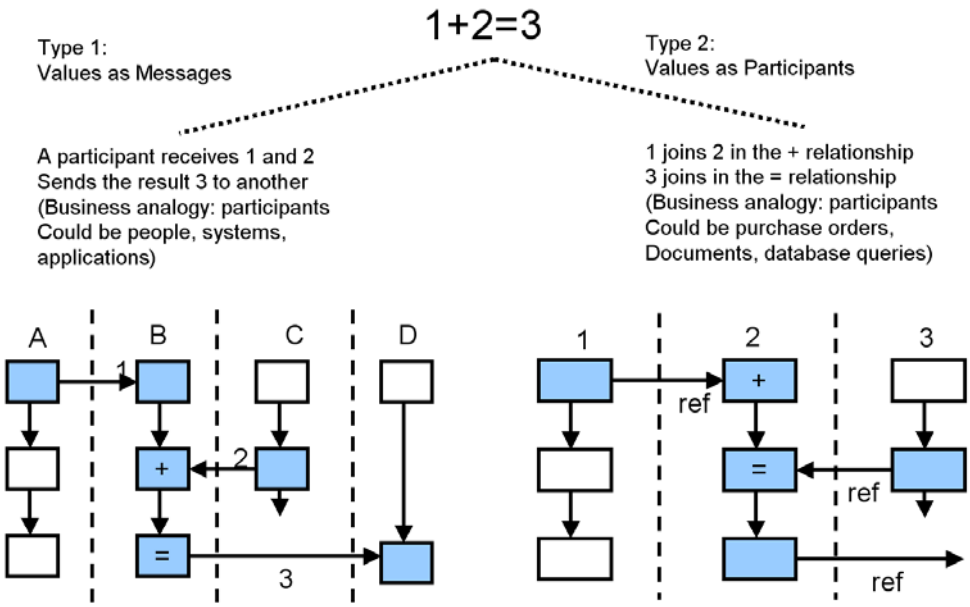

Figure 4 – Two ways to interpret the role of any participant process within a process.

Pi Calculus heralds a future where, as Objects replaced Procedures, we build new Process Oriented Programming (POP) methodologies. In the world of Pi Calculus, every process participant is given a unique Name, and that Name is a central notion of Pi Calculus: the connections between named participants represent the dynamic capabilities and behavior of any given process, at any point in time. Pi Calculus is an algebra in which names represent channels that can act both as transmission medium and as transmitted data. This communication is done on complementary (input and output) channels. Pairs of processes interact with each other by sending and receiving named messages in a synchronized way. The contents of messages are also channels. As a result of such a communication event, the recipient process may now use the received channel for further communication, as in our email example. This feature, the mobility in the system, allows the network "wiring" to change with interaction between the participants. The Pi Calculus provides a framework for the representation, simulation, analysis and verification of mobile communicating systems. Milner has shown that, mathematically, all that we previously understood as computation, and all that we previously understood as communication, can be modeled and understood as the same thing—processes.[3]

Business processes are mobile processes

We observed that the nature of electronic mail could not be easily be implemented on a workflow engine because of the never-ending, infinitely extensible, nature of email communication. Email processes live "in cyberspace", distributed everywhere, and can link and join in new ways, creating new knowledge, just by the passing of email addresses. Despite the difficulty of supporting such process models on workflow

[3] This insight is leading some researchers to consider the possibility of computer hardware based on this new foundation, which could have immense implications for the design of future parallel processing and communications systems.

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

systems, it turns out to be a trivial endeavour using the Pi Calculus.

In its most elementary form, such an email process consists of three participant processes: sender, receiver and address book. Whereas the first two processes are obvious, the address book is the critical third participant. It is this 'process' that allows email addresses to be collected and to flow easily between recipients, therefore allowing the thread of conversation to evolve. As we contemplate this way of thinking about email, we might ask: where is the email message in the process? Here is the noteworthy part … the process *is* the email message. Instead of regarding the message as a document in a flow, the message thread is the flow, or, in other words, the process. The whole email conversation—the process— evolves, not because of the flow of messages or documents among recipients, but because of the automation of email address exchanges.[4]

And here is another noteworthy part … the email threads (actually evolved instances of the email process design consisting of sender, recipient and address book swimlanes) could themselves participate as processes in other processes. This would allow, for example, email processes to participate seamlessly in a supply chain process, or any other process. Rather than the different types of processes, such as email, supply chain, workflow and so on, being considered so special that they must be supported on different technology engines, Pi calculus allows us to treat them as merely different manifestations of the same underlying semantic, and therefore to be supported by a generic technology, the BPMS.

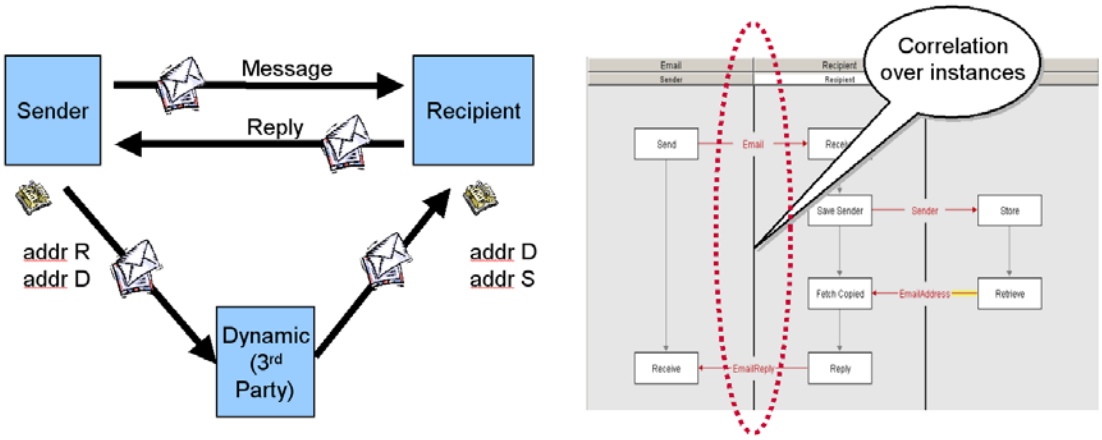

Figure 5 – The essence of how electronic mail works can be modelled as three participants: sender, recipient and address book. As instances of this process execute (or "proceed") each instance gains participants (recipients identified by their email address), illustrating the mobile nature of email.

Whereas previous technology paradigms address only pieces of the business process, the Pi Calculus provides a representation that forms the basis for higher-level process idioms that combine elements across a wide variety of process semantics, including:

[4] One of the earliest applications of the Pi Calculus was to gain greater understanding of Internet protocols.

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

- Automational, eliminating human labor from a process
- Informational, capturing process information for purposes of understanding
- Sequential, changing process sequence, or enabling parallelism
- Tracking, closely monitoring process status and participants
- Analytical, improving analysis of information and decision-making across processes
- Geographical, coordinating processes across distances
- Integrative, consolidating and integrating sub-processes and tasks
- Intellectual, the process of capturing and distributing intellectual assets
- Disintermediating, eliminating intermediaries from a process
- Computational, performing calculations as part of a distributed process
- Collaborative, allowing participants to manage sets of shared work processes
- Compositional, building new processes from elementary reusable process patterns

Most real world processes in business, and particularly those that offer the most value in terms of their optimization and improvement, comprise several of these process types, in combination. Indeed, as we speak to people in business about the processes they care about (new processes they need, existing processes they wish to change, or processes they wish to understand better), the more precise they are about the design of the processes they seek. As they progressively elaborate their requirements, the extent and complexity of the process design grows ever more complex. Numerous systems, employees, partners and machines are required to participate in the process. Not only this, they are required to interact in very sophisticated ways with numerous and constantly changing links. The resulting "to be" process design, often covers a significant percentage of the firm's value chain. It seems more and more unlikely that such a process can be implemented through flows of work organized within a workflow model. Rather, it is more natural to think of interactive design among participants.

Pi calculus, through its distributed and inclusive model of process representation and execution, allows us to build new IT systems that avoid the stovepipe systems of the past that only support discrete business functions scattered across the value chain. Those who have examined this approach to process modeling in practice, remark that not only can Pi Calculus contribute to a more rigorous specification of processes in general and business processes in particular, but also that the modeling of participants as communicating processes is very natural and immediate. This is the reason behind business analysts' common remark that they "find swimlane diagrams easier to use than UML". Business people in particular "get" these types of model very quickly. Give them user-friendly BPMS tools, and they can be empowered to create their own processes, or at least, contribute much more effectively in a multi-disciplinary process re-engineering and systems development effort.

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

Process participants correspond to the way organizations are structured and how people, and computers, work together to achieve goals. Based on the Pi Calculus, we can create technologies that automate the *white space* between all the participants in a process, and, as if out of thin air, the process emerges. Pi Calculus governs the protocols between participants in processes, which are themselves processes. The process *design* defines the protocol, and the process *instance* is the observed behavior of the process.

Workflow is just a process—it can be, but need not be, an engine

Pi Calculus can be used to model any process, including how workflow works, for as it turns out, workflow itself is just a process. Its patterns can be constructed out of Pi Calculus primitives. Workflow management systems allow for the modeling of any workflow because they include workflow engines. A BPMS allows for the modeling of the *meta-model* of how workflows work because it contains an implementation of the Pi Calculus. The text from the product description of Intalio Inc., a company that provides the first standards-based, fully transactional BPMS elaborates on the notion, "Intalio provides a single methodology to model any type of process from the business perspective, automated, manual or workflow. In fact, workflow is just a process to Intalio and infinitely extensible using the Intalio|n³ Designer. Workflow no longer has to be configured and hard coded by IT but can now be changed ad-hoc as the business requirements dictate without the rigor of traditional packaged application approaches."

Once a process meta-model has been defined on the BPMS, such as the way electronic mail works, or the way workflow works, or the way a supply chain works, or change management, or product lifecycle management, etc … it is simply another process that can participate in the design of any other process. But how do we define the way workflow works using these ideas? It turns out that workflow is just a set of participants interacting together, another Pi Calculus process.

The participants in workflow are 'processes' such as activity, task, case, resource, task handler, resource handler, task list and directory. Most readers knowledgeable about workflow systems will not relate to these entities as 'processes,' yet that is precisely what they are when you look at them from Milner's perspective. The whole area is quite confusing since the workflow community and BPMS developers use similar terms for quite different purposes.

Workflow terminology relies heavily on the concept of an *activity*. A workflow activity is used to describe a human or system interaction in the process, for example, the handling of a work item or case. But there is no difference between simple/atomic interactions and more complex interactions. Using BPMS terminology, an *activity* represents a step, or state transition, in a process that cleanly separates control-flow and data-flow, two concepts that are often conflated (mixed together semantically speaking) in non-Pi Calculus based technologies. Thus, when trying to compare WFMS and BPMS one has to recognise that the same terms are being used in different contexts, and really have nothing to do with each other.

This confusion is at the heart of the complexity of trying to compare workflow-derived languages such the WfMC's Process Definition Language, XPDL, with

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

languages such as BPMI.org's Business Process Modeling Language (BPML).[5] Comparing language dialects, XML tag by XML tag, does not get to the heart of differences between WFMS and BPMS. How can it, like is not being compared with like. To make matters worse, in Pi Calculus, remember that everything is a process—including the activities. On a BPMS, a workflow 'activity' may be modelled as a process in its own right. Using Pi Calculus, we *can* model a workflow activity as workflow practitioners understand it, but the way we would achieve this is *not* to use the 'activity' as defined in the specification of process modeling languages such as BPML.

This perspective, where everything is a process, is the way BPMI.org views the world as it develops BPMS technologies and defines process-modeling and query languages. It is a bottom-up approach, in which higher-level processes are constructed from very elementary (low level) Pi Calculus-like processes. It is this approach that can be used to define a BPMS Reference Architecture. The conceptual centre (or 'first-class citizen') of such an architecture is, at all levels, the process.

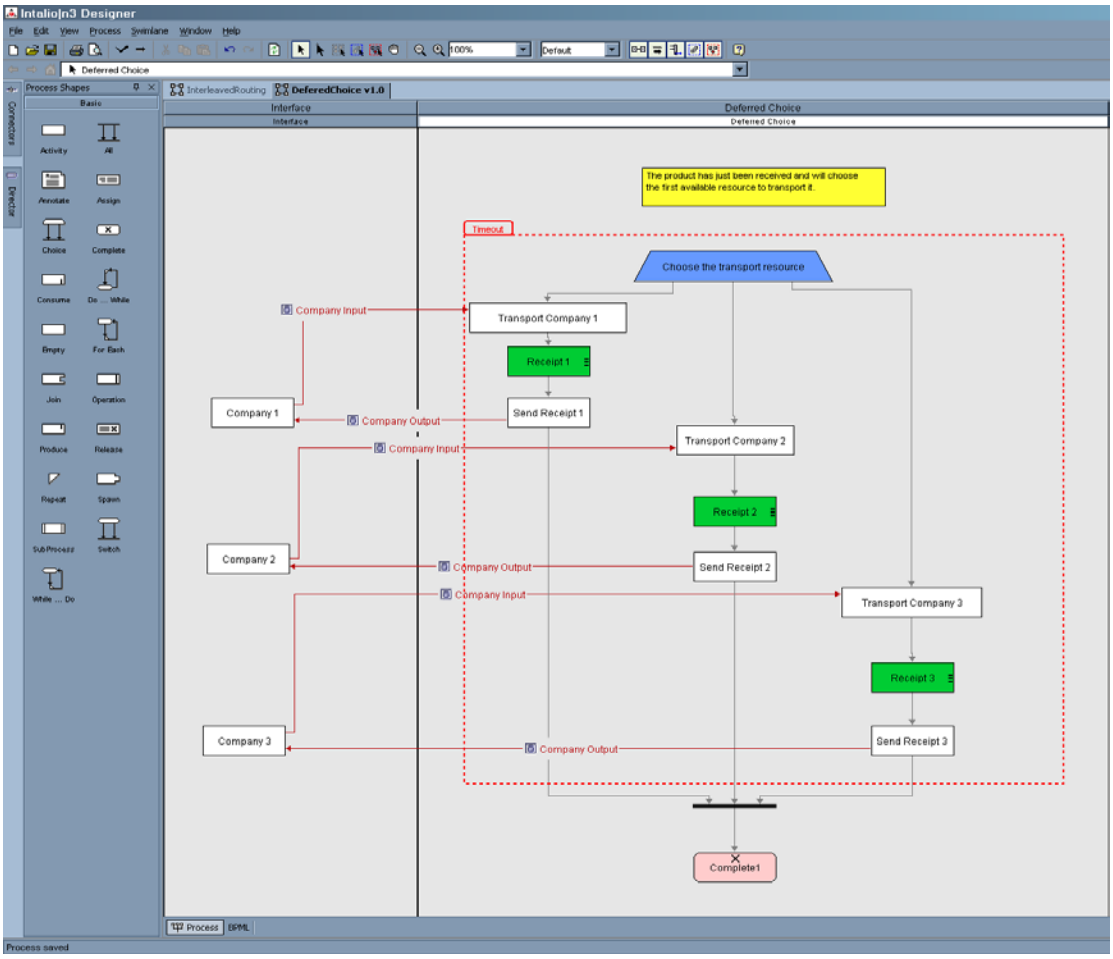

Figure 6 – The workflow pattern **Deferred Choice** modelled in BPML. See Ref [1]. Screenshot Copyright © Intalio Inc. 2003.

[5] Attempts have been made, but reached incorrect conclusions. Rather, it would be more profitable to try modeling XPDL using BPML.

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

Although it is clearly non-trivial to model all workflow semantics using the Pi Calculus, it is quite feasible, and work to date has not revealed a workflow pattern (as identified by workflow theorists) that cannot be modelled using BPML. BPMS products such as Intalio|n³ include workflow functionality, not by integrating a separate workflow engine, but by modeling the required behaviours using Pi Calculus. Pi Calculus is more fundamental than the semantics provided by typical workflow engines because it can model a far greater number of processes and patterns, including workflow patterns, even workflow engines themselves. This can be clarified by realizing that it is not possible to write a workflow engine using a workflow engine! This is why we consider languages such as BPML to be more fundamental than higher-level workflow languages such as the WfMC's XPDL. In fact, it should be possible to model XPDL using BPML and deploy it on a BPMS, just as Intalio has shown how it is possible to model all workflow patterns using BPML [Ref 1].

Example: Deferred Choice is defined as a point in the workflow process where one of several branches is chosen. In contrast to the well understood XOR-split, the choice is not made explicitly (e.g., based on data or a decision) but several alternatives are offered to the environment. However, in contrast to the AND-split, only one of the alternatives is executed. This means that once the environment activates one of the branches the other alternative branches are withdrawn. It is important to note that the choice is delayed until the processing in one of the alternative branches is actually started, i.e., the moment of choice is as late as possible. The BPML for this process is:

```
<?xml version="1.0" encoding="UTF-8"?>
<package version="1.0">
   <process name="DeferredChoice">
      <sequence>
         <choice name="Choice1">
            <completeBy duration="&amp;quot;PT1M&amp;quot;"/>
            <onException>
               <assign name="Assign1" target="receipt"
                select="&amp;quot;Time out received&amp;quot;"
                append="false"
                xmlns:xsd=http://www.w3.org/2001/XMLSchema/>
            </onException>
                  <sequence name="sequence1">
               <consume>
                  ...
               </consume>
                  ...
            </sequence>
            <sequence name="sequence2">
               <consume>
                  ...
               </consume>
            </sequence>
         </choice>
      </sequence>
   </process>
</package>
```

A workflow engine engrains the workflow process meta-model into its software code, just as an ERP system engrains its processes into software code (although it keeps the data model separate using an RDBMS). As a result, workflow systems, ERP systems and most packaged applications cannot be easily changed at the level of process meta-model. They are limited to expressing the processes they were designed to

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

express, nothing more, and nothing less. These are the underlying reasons why some have observed the extreme rigidity of ERP applications (illusion: wet concrete before installation, dry concrete after installation) particularly after they have been heavily customized, as they must be if they are to support the unique competitive processes of G2000 firms. Unless the ERP architecture itself evolves to support BPMS (and with that a more fundamental model of workflow as described in this paper) the view that IT has become a commodity, offering no competitive advantage, will become more prevalent. As it stands, ERP can be counter-productive, since companies are required to 'copy cat' one another and deploy precisely the same processes offered to them by their ERP vendor.

Many workflow systems are based on classical Petri Nets[6] that, as Marc Förster of the Hasso Plattner Institute for Software Systems Engineering explains, "… have been used for a much longer time than algebraic methods like Pi-calculus to formally model processes and in many aspects possess similar expressive strength. Compared to Pi-calculus expressions though, they have a fixed connection structure and thus lack the possibility of dynamically changing their behaviour by interaction. Since data flowing between processes may, in Pi-calculus, represent whole processes itself such dynamics can be expressed." [Ref 2]

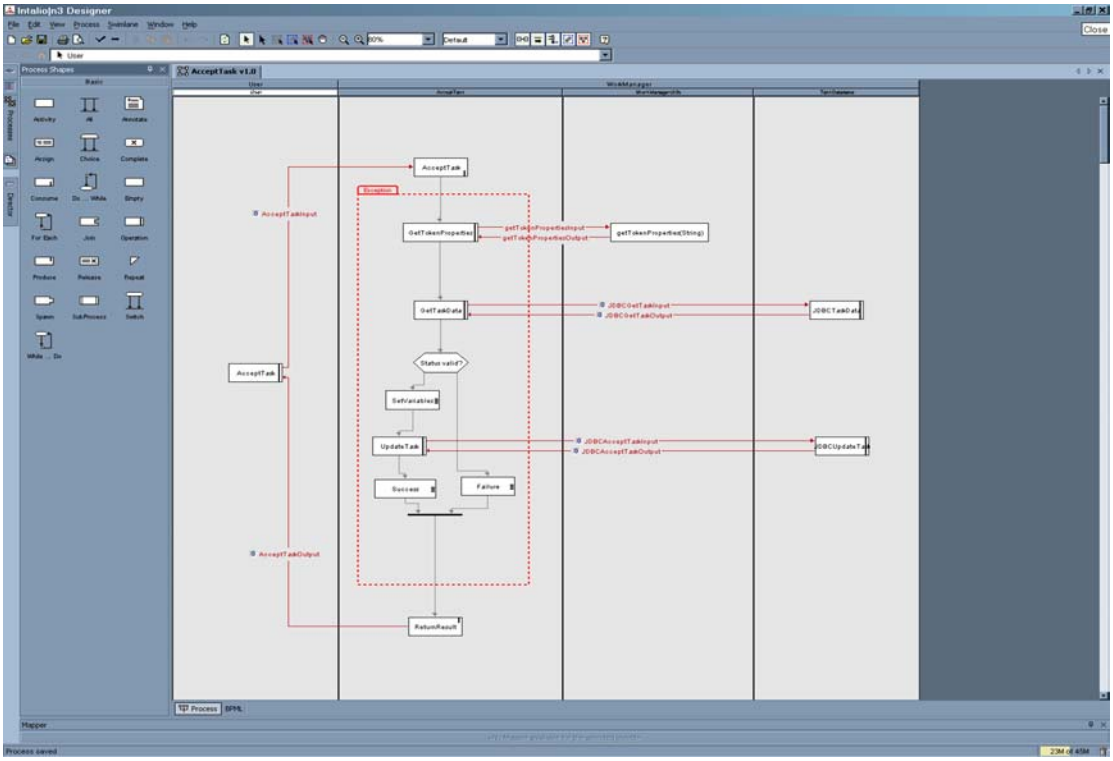

Figure 7 – The **Accept Task** process, a component of a workflow manager, modelled and executable in Intalio|n3 BPMS, and reusable as a process within other end-to-end processes. Screenshot Copyright © Intalio Inc. 2003.

[6] Petri Nets are a major source of inspiration for developers of workflow management systems, and many workflow engines are built on principles derived from theories of Petri Nets. Over the years, Petri Nets have been extended to support the needs of both theorists and practitioners, for example the Color, Time and Hierarchical extensions.

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

A new foundation for processes via a process virtual machine

Pi Calculus is a universal mathematical language for processes. In the Pi Calculus, as we have explained, everything is a process, even lowly integers such as 1 and computations such as 1+2=3. We can look at this in two ways: either 1, 2 and 3 are participants in the process of computation, or they are interactions between variables (themselves processes) that yield the result 3. Lists, queues, data stores, procedures, communicating threads, everything that exists in computing today, can be modelled using Pi Calculus. It may not always be easy, but it can be done, and these processes are then reusable within the design of other processes.

This surprising insight, the result of twenty years of computer science, may be interesting in itself, but its significance would stop there were it not for the fact that Milner's concepts have been implemented in practice. Initially, computer scientists began developing experimental programming languages based on Pi Calculus, such as PICT and Join. Join extends the Pi Calculus with the Join construct. Join was then the inspiration for more recent efforts and the development of industrial-strength languages such as BPML. BPML was in turn used to develop enterprise-scale BPMS products such as Intalio|n3.

A BPMS should include a *process virtual machine.*[7] This single runtime provides the foundation for process execution, and such virtual machines are based on the Pi Calculus. Again, this would be interesting but not significant, were it not for the fact that early implementations of BPMS process engines are able to massively scale, allowing processes to be deployed that rival or exceed those supported today in ERP software. One analyst predicts that such implementations can exceed the performance of existing software systems because existing software relies on separate computational and communicating threads, implemented within heavy-weight messaging and transaction processing monitors, with the associated overheads arising from the layers upon layers of legacy code and baggage that comes from decades of product evolution and extension, mostly to make up for deficiencies in the legacy itself. By contrast, Pi Calculus engines provide the illusion of communication between participants with no requirement for message passing. Additionally, transactional semantics are inherent in the model, which can ultimately avoid traditional transaction processing altogether.[8] The true benefits of process systems in terms of resource utilisation will emerge over the next few years as practical experience grows. They will eventually break free of legacy systems and demonstrate their value in their own right, as occurred with previous generations of new technologies.

The BPMS can support the most complex, dynamic and extended processes. Such

---

[7] A process virtual machine can be implemented within the existing computing architecture of Java and J2EE within a container. In the future, process virtual machines will rival today's Java virtual machines in terms of establishing a new platform for Process-Oriented Computing.

[8] Today of course, BPMS products are deployed over existing messaging, transaction processing and application connectors, leveraging previous investments. Many processes deployed on a BPMS today will include participants (represented in swimlanes) that are in reality models of the underlying systems. This is called process projection.

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

processes are persistent, reliable and transactional. They are in fact a new class of business asset—dynamic processes as opposed to static documents. They can be treated as documents just as we treat Word processing documents or electronic forms today, but there the similarity ends. While metadata may be used by workflow-driven document management systems to keep track of the use of documents, process documents inherently encode the *lifecycle* of the information they manage, from creation to disposal. Within a process, all data is encoded in the context of its use—past, present and future process design. A BPMS manages this new class of digital content and can therefore be considered to be the platform upon which the IT industry will build the next generation of enterprise business applications, including the next generations of ERP, document management, workflow, content and knowledge management.

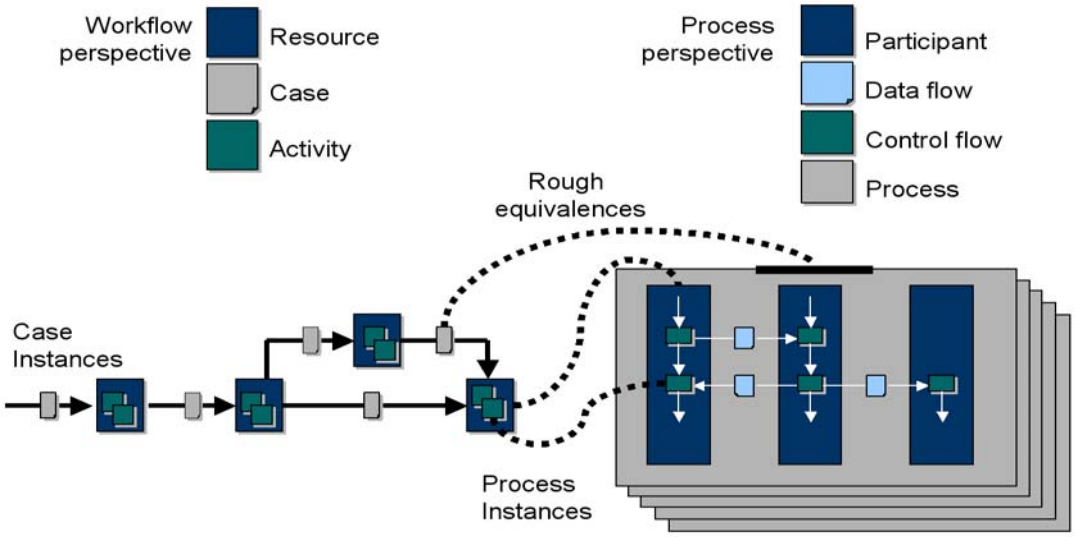

Figure 8 – In BPM, the notion of a workflow case, or document, is the end-to-end process, creating a new form of digital content (XML BPML instances). For example, in BPM, an insurance claim, or a medical record, actually *is the process*, as opposed to the static document *being processed*. In BPM, different process instances (the new cases/documents) can evolve differently over their lifetime.

The Business Process Management System

Using a BPMS, a business team can develop a process model, from the highest level of abstraction down to the most intricate details. It can be deployed directly, effectively creating an "instant" new business application. This capability might be disregarded as a cheap trick were it not for the fact that the BPMS can reuse the processes engrained in existing IT systems in which companies have heavily invested.

Process modeling, using a BPMS, is not always about creating new processes. It is often about discovering, re-describing and re-casting existing processes so that they can be used in new contexts and in combination with other processes. This is variously called process consolidation, process digitization or process fusion. It is a fundamental capability of a BPMS. Just as relational data management systems supported the aggregation of business data and the creation of application and enterprise data models, a BPMS achieves the same for business processes. As RDBMS created new value from data, BPMS will create new value from process.

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

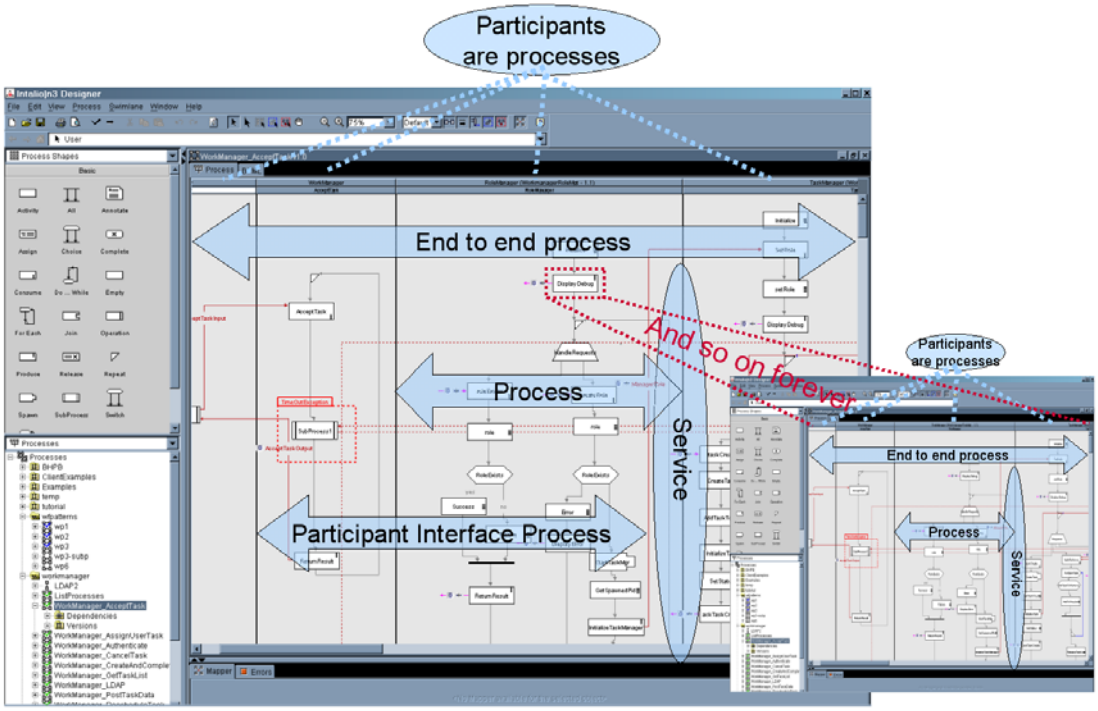

Figure 9 – Processes are reusable, compositional and can be nested.
The principle allows bottom up and top down process design.

A BPMS does not 'integrate' applications and Web services as many workflow solutions and EAI do. That approach only creates aligned data and some workflow control over messaging. By contrast, a BPMS assists in the direct reuse of existing investments in IT processes by consolidating them within a process-oriented architecture (POA). To do this it uses a technique called *projection.* The BPMS exposes underlying systems and the process engrained within them based on their meta-model of processing and creates, dynamically, BPML processes. These can be then be combined with others. Because of the Pi Calculus we can view business processes as a new abstract data type. This means we can persist them as data records in a BPMS *process base,* a database of process records. Like stored information within the thread of email, the process base contains the past, present and alternative futures (via simulation) of the stored process.

One way to understand the BPMS capability is to think about leveraging existing applications as real-time systems of records, allowing new processes to be externalized into what Ismael Ghalimi (co-chair of BPMI.org and Chief Strategy Officer at Intalio Inc.) describes as a transactional system of actions. The implications to application maintenance, extension and product-level integration, are clear.

The BPMS creates the opportunity for *process manufacturing* and mass-customisation on a scale previously unimaginable. The BPMS provides businesses with the capability to conceive and put processes directly into operations without distortion through direct execution of process models or process model variants created on the fly, in real-time. The BPMS puts process at the heart of enterprise architecture and will ultimately have an impact on IT and business management similar to computer-aided

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

design and computer-integrated manufacturing did when 3D product and component models were put at the heart of the new CAD/CAM toolsets.[9] In the past IT has automated the business; in the future the CAD/CAM-like BPMS will automate IT.

With the BPMS platform in place, the traditional divide between business and IT implementation is eradicated because the BPMS creates a shared language of process and because consolidated end-to-end process models are deployable with no intervening software development. The ability of the BPMS to project existing processes, from systems such as ERP, CRM, SCM and other legacy systems, works because the Pi Calculus is able to express any pre-existing combination of computing and communicating participants, from the details of code in one application, to massively-distributed processes occurring between computers over networks.

A world of processes

So far in this article, and despite lots of talk about Pi Calculus, we have given only one example of a business process that workflow just cannot support, that being electronic mail. While you might be convinced that workflow engines cannot implement electronic mail, email is only one example, and a rather trivial one. To provide a convincing argument for the necessity for Pi Calculus-based technologies, we need to demonstrate other process models problematic for workflow engines to implement. They will fall into two camps. Either they will be impossible to important, like email, or would require so many workarounds to be implemented in the WFMS that developing the end solution would be uneconomical or difficult to maintain. But before we abandon the email example and move to others, it is worth reflecting on the fact that mobility, as Milner has shown in numerous cases, turns out to be rather fundamental.

Many companies are looking to enhance email to provide more structured collaboration, and more significantly, to understand the patterns of collaboration and the lifecycle of particular collaborative activities in areas such as product design, marketing, sales and the management of supply chains. Indeed, those companies that have tried to apply workflow in these application areas have not found it an ideal way to think about those problems. While email may be too ad-hoc, workflow is too prescriptive. The answer is not something in the middle, but something of a different character: mobility in all processes, from email, to workflow, to integration, to ERP. Workflow's flow model just seems unnatural for many processes.

One case is the area of *coordination and negotiation.* Such processes extend workflow in ways that require team members, or systems, to commit to work before accepting it and may require those accepting work to have negotiated commitments with others before doing so. These *loops of work,* the commitment and negotiation between individuals and teams, within and between enterprises, must be managed and collapsed back to originators of work before work can proceed. One software

[9] It is estimated that the introduction of CAD/CAM/CIM in manufacturing enhanced the productivity of those firms that adopted it by two or three orders of magnitude, and opened the door to mass-customization.

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

company, Action Technologies, specialises in an engine, not a workflow engine, to support such commitment and negotiation processes. It is based on the early groundbreaking work of Terry Winograd and Fernando Flores in *Understanding Computers and Cognition*.

The approach adopted by Action Technologies has been found to be effective in a wide range of process domains, for example, customer service in complex industries, where coordinated teamwork is required. Once again, the extended, mobile and dynamic aspects of these looped processes cannot be represented using workflow. In fact, one company that wanted to implement coordination and negotiation went to the trouble of asking their preferred workflow vendor whether they could provide the same functionality. The answer was that two years of development would be required. Thus, while workflow engines can be extended outside of their core capabilities, this is hardly an option for many processes. As the company found out, it was the coordination and negotiation processes, not a workflow, which helped the company solve its customer service objectives. Had the company deployed a classical workflow engine, it is unlikely that those objectives would have been achieved. By contrast, we expect that the coordination and negotiation processes can be readily modelled and deployed using the Pi Calculus and BPML approach.

There are thousands of possible business processes that have nothing to do with the standard model of workflow inherent in existing workflow products. We are finding more and more processes where modeling using workflow is either awkward, requires workarounds or is simply impossible. Change management and product lifecycle management is another. Processes change your perspective, from viewing a change request as a document (processed by a workflow) towards viewing it as a process in its own right (mobile through participants). Even in cases where a workaround on a workflow solution might work, the workflow-based solution might not provide the full coverage of the process lifecycle, end-to-end, and therefore its improvement over time.

There are also applications for BPMS where WFMS does not apply at all. For example, while many businesses seek to reduce the cycle time of a process, it is also critical in some contexts to reduce the time taken to create a new process, or to customize an existing process—what we call *process manufacturing*. While workflow engines can accelerate the creation and customization of the workflow in a process, it cannot extend to the whole process. Processes such as order to cash in the fashion industry, supply chain in fast moving consumer goods, product lifecycle and change management in aerospace, sales campaign management in chemicals and clinical processes in healthcare create requirements for end-to-end process models that span multiple systems, technologies, information sources, employees and business partners. These processes are long-lived, persistent and unique to individual companies in two ways. First, with respect of the basis of competition, and second, with respect of the linkage between the process design and the myriad systems and business practices that must be projected in order to participate. This is not a workflow problem. It is a process management problem and it requires a new platform for process representation and execution—the BPMS. The process models for consolidating the lifecycle of end-to-end enterprise processes don't look or feel anything like workflow models. Coercing them to do so, even in cases where it is possible, is counter-productive.

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

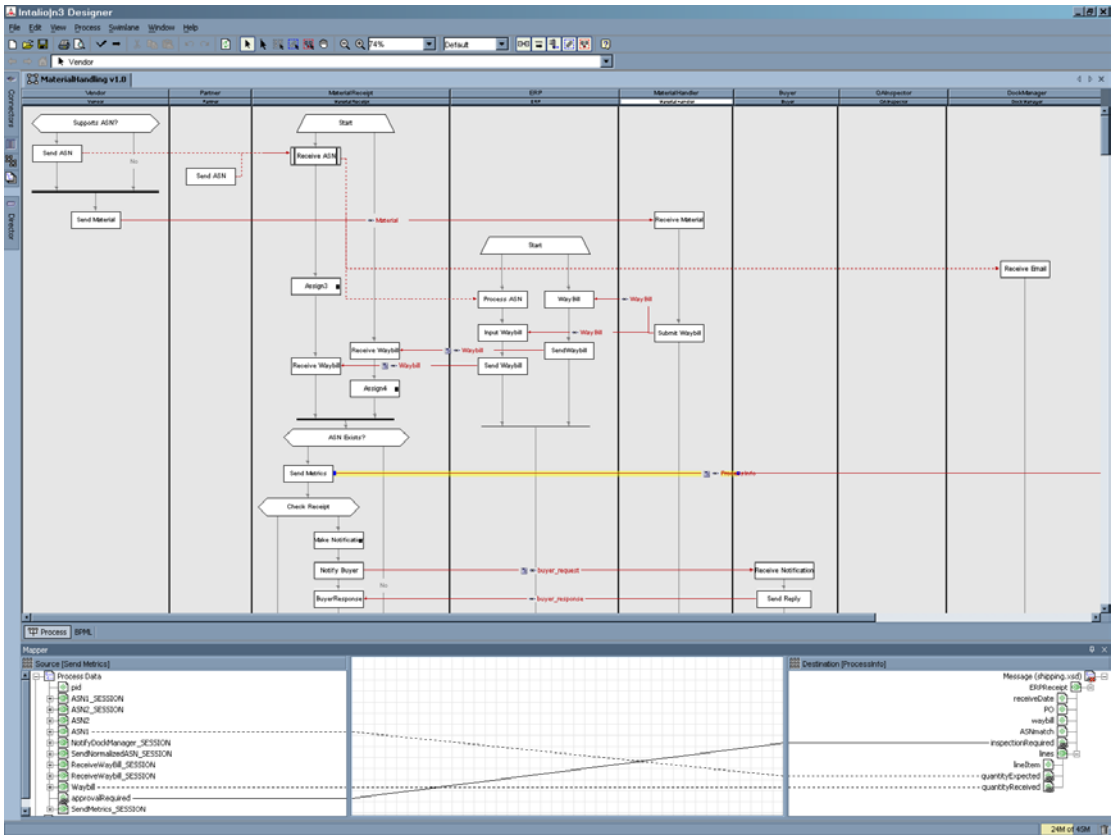

Figure 10 – An example of a Materials Handling Process, whose participant processes (swimlanes) include business roles such as Vendor, Partner, Buyer, Inspector, Dock Manager and QA Inspector, as well as 'white space' processes such as Materials Receipt and Materials Handler, and participating business systems, such as ERP. Instances of this process represent the life history of the handling of certain materials in the enterprise. Screenshot Copyright © Intalio Inc. 2003.

Very often it is the nature of the processes themselves that present severe, even insurmountable, challenges for workflow technologies. For example, consultants who have examined clinical processes have uncovered requirements far from classical automation. In workflow it is typical to have few process designs, with many instances. In clinical practice there are many processes *and* many instances. In addition, due to unique patient history, symptoms and treatment needs during life, a lot of customization pointing towards orchestration between many participants, and not just automation along a flow. What is need in healthcare is a process management capability where the process, corresponding to patients, doctors and medical procedures, can be customized before the patient makes their journey through the healthcare system, and later the possibility to modify the remaining part of the process, to switch the patient over to a new process because of discoveries during diagnosis or treatment.

Business process management provides a user interface, without workflow

We have discussed how a Pi Calculus based BPMS can model the semantics of workflow. We have assumed, during this discussion, that the user interface to the

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

process will look and feel much like today's task-list oriented workflow systems. In fact, BPMS opens the door to completely new ways to interact with processes. Rather than being a cog in the wheel, driven by a task list, workers can fully engage with processes using a BPMS.

It is possible to model the user as a participant in the process. The boundary of communication between this user (a swimlane in the process model) and the other swimlanes creates multiple intervention points for the user in the process, and opens the possibility for the automatic generation of a skeletal Web-based user interface. By adorning this with user interface widgets of various kinds, sophisticated user interface screens can be developed, and the process paradigm can be extended, right down to the level of Web page fields that interact in the process and with the user. This can create a much richer user interface than a typical workflow task list, although such a task list may be needed in many circumstances. It too is a process. As we have shown, lists are themselves nothing more than processes.

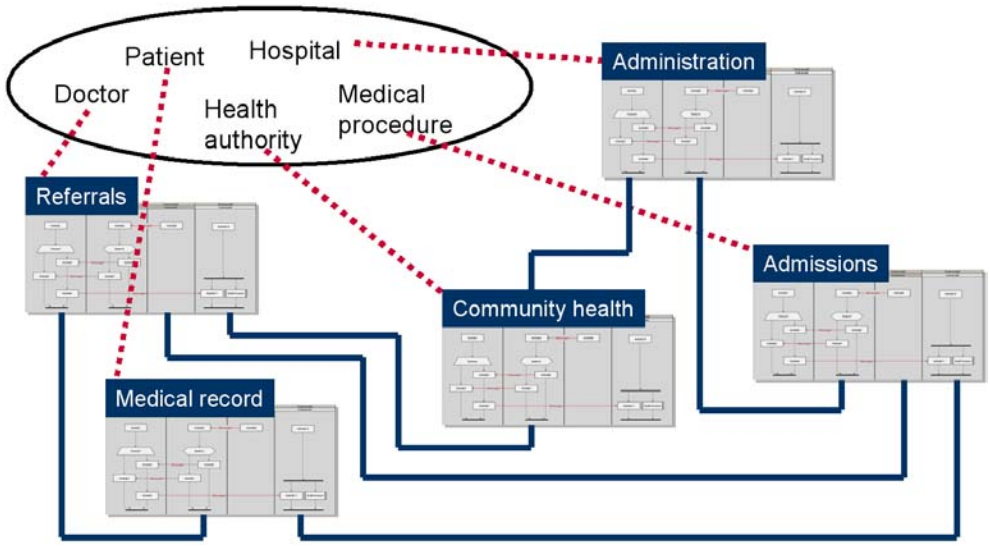

Figure 11 – Example, a medical records handling system modelled as end to end processes, whose lifecycle history represents the unique medical history (past, present, future) of the individual patient participating in the medical system. Process Systems create a new way to model business. Processes are the new 'Objects' and entities that business analysts seek are just Processes that participate, one within another.

Since any end-to-end process model (see Figure 11) will probably contain swimlanes (processes) for many different business user roles, all of the required user interfaces can be generated at once, allowing multiple human interface points into the process, depending upon role and access rights. These capabilities are already available in today's BPMS products and will improve significantly over the coming months and years.

Workflow won't just *go away*

The rise of BPMS, and the power of Pi calculus, does not mean that the market for WFMS is going away any time soon. Workflow management is a useful form of process management. Advocates of workflow-based technologies will continue to

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

promote their products under the banner of BPM, yet it is important for businesses to understand the differences between workflow and process foundations. BPM is a new journey, not just for vendors of BPMS products, but also for vendors of other technologies (e.g., EAI, workflow, business rules and application servers).

Many workflow vendors, even some rules vendors, adopted the marketing term "business process management" because it expresses a current, and urgent, business need, and is therefore more saleable than the older term 'workflow', which recalls, for some CIOs, disappointments from the past. For this reason, workflow vendors have extended their WFMS with additional capabilities, for example EAI and rules, adding to the utility of the WFMS. Analysts, who are hell-bent on categorising market trends using three letter acronyms, define "BPM" as this *bundling* and *integration* of related capabilities, and while this is one interpretation of the way the market is developing, it hardly gives companies an understanding of the shift toward process-oriented systems based on the Pi calculus. And the trend to use the term "BPM" is not limited to workflow vendors.

Companies that provide *workflow-driven applications* or *process-configured code generators* justify their use of the term "BPM" because of features specific to their methodology and implementation. For example, one vendor provides powerful capabilities to monitor the performance of workflow in ways that let business people make changes directly, literally with no involvement by IT technicians. The users themselves align the business workflows to their needs as they arise. Another vendor uses workflow diagrams to hot-configure EAI processes.

Each of these "BPM" technologies has value in the marketplace. However, end users need to be aware that products marketed under the moniker "BPM" have less in common with each other than use of the "BPM" moniker might indicate. In one case, two vendors who sold a "BPM" product found, after understanding each other's technology in more depth, that they were actually complementary and are now willing to work together. Vendors such as these are coming to recognize that the Pi Calculus and the BPMS platform is a unifying architecture in which they are able to work together, in practical terms as opposed to marketing terms!

How many engines can one company support?

> *The application of contemporary workflow management systems is not always able to cope with ill-defined and unstructured environments. In practice, workflow technology often lacks flexibility, because it is trapped in a control flow paradigm. Workflows should not be driven by pre-specified control-flows but should be data- or information driven. -- Ijme Schilstra, BPM'03, Eindhoven*

Pi Calculus based technologies will not be used to fully re-produce the functionality of email systems, full-featured workflow systems or sophisticated collaboration products at this stage in the market. These existing technologies will evolve, while along side, the BPMS will emerge as a new category. In many cases existing workflow products may provide "good enough" process management. On the downside, unless we move forward to extend the process paradigm into the broader area of processes (including its extension as the basis of business applications such as ERP) each enterprise automation requirement will forever require a separate engine. What's all

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

too common is the creation of a "process hyper-tier" where multiple technologies are marshalled or spliced together to support business process management. But here is the catch. Each technology requires it's own engine—a workflow engine, a rules engine, an EAI engine and so on. But it gets even worse when the need for industry specific collaboration and compliance are considered—a HIPPA engine, an ebXML engine, a Sarbanes-Oxley engine, an EDI engine, a RosettaNet engine, a Six Sigma engine ... all of which may require differing workflow and rules engines pre-packaged by solutions vendors. With how many engines can one firm cope? Is end-to-end process management (discovery, design, deployment, execution, operations, optimization and analysis) viable across an IT infrastructure where processes are broken up into little pieces corresponding to different engines, both semantically and piecemeal? Can a universal process engine help solve this engines "arms race?"

Today, the IT function is bogged down in a host of integration challenges[10] between a host of different systems and engines, including, numerous applications, workflow systems, integration hubs, collaboration tools, business-to-business exchanges and others. A typical G2000 firm has, literally, several hundred, or in some cases, thousands of IT systems. Overlay these with the *end-to-end* "value chain" business processes that cross multiple companies and the situation is ultimately untenable, giving rise to the current market uncertainty about the value of IT.[11] Add to this the following requirements and realities:

- Competition: A continual focus upon *end-to-end* process improvement;
- Change: An uncertain, constantly changing, business climate;
- Globalization: Accelerating the requirement to work ever more closely with far-flung partners;
- Regulation: New legal requirements for transparency across *end-to-end* processes.

Taking these factors into account, one can quickly see extreme complexity in process infrastructures that, if taken on piecemeal, are not only costly to acquire, build and maintain, but also quite inflexible with regard to the natural evolution of the business. The BPMS is a pragmatic step forward in meeting these challenges.

A BPMS, based on Pi Calculus, can represent any process in other technologies, and can create consolidated *end-to-end* process models that can be managed with a single, holistic process engine. In one case, a BPMS was proposed to create an end-to-end process across four different WFM systems owned by different business units. Each workflow system had been procured for what, at the time, seemed good reasons. Different business units had different "feature and function" requirements, and each found a workflow engine to meet their needs. However, each engine was from a different workflow vendor and embodied different workflow semantics. When the requirement for new end-to-end processes arose from evolving customer and

[10] Estimates vary from 30% to 60% of IT budget.

[11] www.bpm3.com/hbr/ (Nicholas Carr's "IT Doesn't Matter" and industry responses)

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

marketplace needs, it took a serious internal debate over which workflow engine should implement the new end-to-end process. Finally, a BPMS was proposed. Even if one WFM system had been chosen, could it provide the flexibility to represent and manage the full lifecycle of the end-to-end process envisaged, or was the requirement for the new process a manifestation of deeper underlying issues, the failure of existing technologies to create and manage consolidated processes, and the tendency of existing technologies to create stovepipe IT systems? While this is an extreme example, it illustrates the dilemma facing CIOs and Process Officers as they consider how best to provision new processes.

Whether integrating workflow engines, integrating workflow with ERP with CRM with SCM with PLM with partners with legacy with … and so forth … the BPMS represents a powerful *transition capability,* allowing businesses to leverage existing assets while simultaneously moving to a process-oriented architecture (POA). This is important because within a POA, the conceptual centre is the business process itself, the focus of management attention.

In the same way that the RDBMS, based on the relational model of data management, replaced disparate hierarchical and network-oriented databases, we believe BPMS will replace multiple approaches to workflow. Yet workflow unification alone is an insufficient capability to secure the future of the BPMS. The much broader process-oriented requirements of an enterprise are why the Pi Calculus is so important. A process does not replace just workflow, but encompasses every element of what today we call "applications." The BPMS, based as it is on a fundamentally different underpinning and approach to that of workflow, can unify all existing IT systems to one degree or another. In short, business processes are much more than the flow of work. The BPMS heralds a change in the IT stack itself, from applications built on a data foundation, toward process management tools built on a process foundation.

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

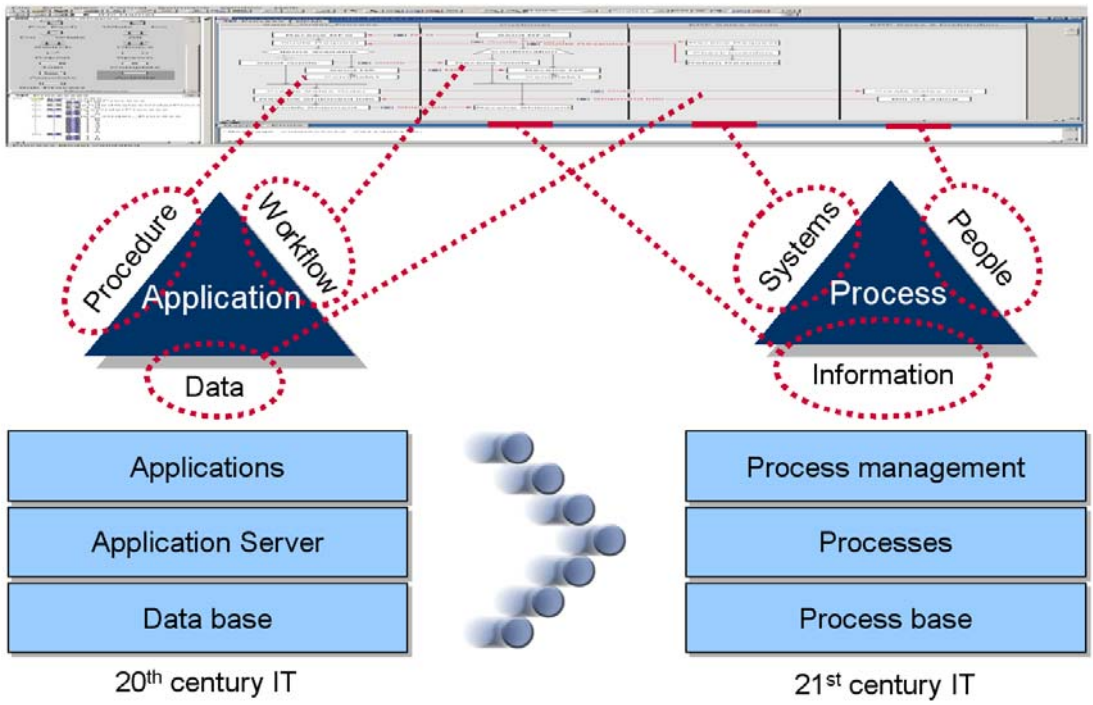

Figure 12 – BPMS heralds a shift from applications to processes, based on their respective technology stacks. Software engineering, and the emerging Model-Driven (MDA) are complementary to the emerging process manufacturing and Design-Driven Architecture (DDA) enabled by BPMS.

A symbiosis between standards-based commodities and new innovations

We can also think of the emergence of the BPMS in terms of unanticipated innovations that it can engender. No one knew what to do with a PC until they saw a spreadsheet. No one knew what to do with Unix until they saw a RDBMS. Today, no one knows what to do with Web services until you show them a BPMS. These symbiotic relationships between standards-based commodities (e.g., PCs, UNIX and Web services) and new innovations (Spreadsheets, RDBMS, BPMS) create new value from IT.

The BPMS is the innovation that makes sense of today's standards-based community platform, Web services—the BPMS has been described as the "killer app" for Web services. The BPMS platform provides a process-oriented architecture (POA) that can be deployed over today's Web services platforms that are, by contrast, service-oriented architectures (SOA). Web services are just fine at exposing the process participants the BPMS can exploit. Web services live in the era before Pi calculus-based technologies. They represent the final standardisation of 20th century technology, and for many businesses that's long overdue. By contrast, the BPMS is a 21st century innovation and ripe for market adoption.

The BPMS makes sense of the past investments in IT by normalising, re-describing and flattening their disparate process models, allowing them to be combined, re-

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

purposed, customized and extended in myriad ways to meet new business needs. The BPMS capitalizes on the fact that IT giants such as IBM and Microsoft are finally standardizing distributed computing using Web services APIs. This is allowing vendors of BPMS products to translate technical details of underlying IT systems into reusable building blocks that can be understood as business processes, and used to create and manage those business processes. The existing IT systems and the services they provide through Web services APIs are like the 3D-component building blocks that CAD/CAM designers assemble to create and manufacture new products.[12]

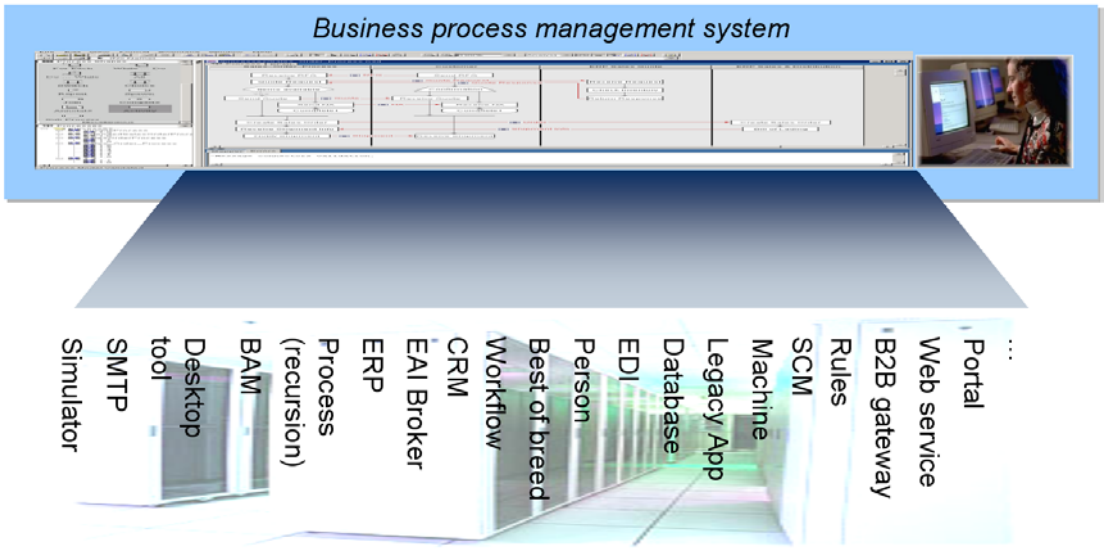

Figure 13 – Consolidation of enterprise processes to create end-to-end process models

The Business Process Management Initiative (BPMI.org)

The eradication of variations in workflow semantics, the extension to the full process lifecycle over existing technologies and the integration of top-down and bottom-up process modeling methods and tools was the task BPMI.org set itself when, in 1999, the co-founders first met together. Now, with the availability of BPMS products already in the market, the following months and years will validate the power of Pi Calculus-based technologies and confirm the BPMS as the platform of choice upon which G2000 companies will build the next generation of business information systems. The significance of the Pi Calculus lies mainly in two areas:

- The fact that it unifies two concepts that were previously thought to be quite different phenomenon, computation and communication;

- The fact that it can be viewed simultaneously as a simple programming language in which processes can be described and, as a mathematical object about which rigorous digital expressions can be proved.

There have been countless attempts over the last two decades to raise the abstraction level of representing processes through one form of data-flow or the other. Some go

[12] Using a BPMS has been described by one business architect as "Lego-Block Systems Integration"

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

under the name of CASE (computer assisted software engineering). BPMS is a not a CASE-based approach. The fundamental problem with CASE and other techniques is that, at some lower level of abstraction, the process elements must decompose to control-flow. The language producer is typically burdened with continuing to add additional parameters to the process elements (the semantic building blocks) to meet expanding developer expectations and, sooner or later, the model collapses on its complexity. The vendor may open the language specification to the developer community at the risk of losing interoperability as the platform splinters into different domains usage through divergent scripting-extension forks. BPMS avoids this problem by including process virtual machine technology that serializes concurrent computation to a single thread, which is executable over existing software and hardware systems. This "bottom up" approach, based on the combination of a process modeling language and process run time, lies at the heart of BPMS. Put simply, the foundation ensures that truly any level of process complexity can be encapsulated. The lines between computation and communication are obviated using BPML and they become one and the same. Activities are just processes, workflows are just processes, applications can be de-composed into processes, and all these processes can be composed and nested infinitely.

Pi calculus, and languages such as BPML, will have a profound impact on how control flow and data flow are formally separated within process models, but then re-unified for the purpose of process management. However, workflow theorists will continue to pursue further developments in the field of adaptive and concurrent workflow.[13] While it is theoretically possible to extend workflow technologies and underpinnings to be more expressive and to provide more flexibility, the first BPMS projects are already being completed by leading companies, pioneering the use of Pi Calculus in the enterprise, as occurred before with the relational model of data. It is indeed these reasons that the unfolding story of a business rEvolution is about BPM, not just a subtle evolution of workflow. Pi calculus underpins the computer science of distributed mobile processes, a paradigm where the business process is the indigenous species.

Now enter the marketplace and the force IT vendors so energetically exert to influence markets to buy their products. As vendors re-position their heritage products as "BPM," and as analysts' magic quadrants fill with names of winners and losers, the battle for the BPM market share (the re-invention of workflow *and* applications) has begun. Along the way, as companies seek true innovations for competitive advantage, the BPMS must prove itself in mission-critical projects, one by one. In the battles for the future, it is indeed the marketplace that always wins—and the marketplace will ultimately demand a unified, holistic engine of process. In a world of Pi Calculus, all processes, including the workflow process, are just that, processes, not engines. Let the games begin!

---

[13] Petri Nets are being extended in a host of ways, including taking over ideas from Events, Agents and Actors. Concurrency within and between Petri Nets is also an active area of research.

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

References

[1] Intalio Inc., "Technical Note: Implementing Workflow Patterns in BPML," V1.0. The paper illustrates how workflow patterns that workflow engines typically cannot support can be modeled and executed in BPML. The paper deals with the complex workflow patterns identified by W.M.P. van der Aalst, being Synchronizing Merge, Deferred Choice, Interleaved Parallel Routing, Multiple Instances without a priori runtime knowledge, Milestone, Cancel Activity and Cancel Case.

[2] Förster, M., "Theory of Business Process Modeling: The Pi Calculus," Hasso Plattner Institute for Software Systems Engineering, Prof.-Dr.-Helmert-Straße 2-3, 14440 Potsdam, Germany

[3] Yet Another Workflow Language, YAWL:
http://www.citi.qut.edu.au/yawl/index.jsp

[4] Collected resources aimed at giving an overview of the Pi Calculus, the work of Robin Milner and the relationship to Business Process Management www.bpm3.com/picalculus

Further reading

[5] Smith and Fingar are co-authors of two books about business processes, *Business Process Management—The Third Wave* (December 2002, Hardback 311 pp) and *IT Doesn't Matter—Business Processes Do.* (July 2003, Paperback 128 pp). They can be previewed at www.bpm3.com

[6] Howard Smith, together with colleagues from Computer Sciences Corporation's Research Services, published the industry's first report on *The Emergence of Business Process Management* in November 2001. It was published by Computer Sciences Corporation Research Services and re-produced by ebizq.net and is still available as a PDF download (90 pp): http://www.cscresearchservices.com/process/bpmreport and http://www.ebizq.net/hot_topics/bpm/features/1763.html

[7] An early perspective on why a Business Process Management System is relevant to Systems Integrators was published by Computer Sciences Corporation in a whitepaper entitled, *A System's Integrators Perspective on Business Process Management, Workflow and EAI,* http://www.fairdene.com/processes/CSC-SI-Perspective-BPM-v1.pdf

[8] Will the Real Business Process Management Systems Please Stand Up—
Separating the Pretenders from the Contenders
http://www.darwinmag.com/read/070103/pretender.html

[9] Business Process Management 101
http://www.ebizq.net/hot_topics/bpm/features/2830.html

[10] Clearing up confusion over what BPM really is
http://www.ebizq.net/topics/eai/features/2225.html

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation

[11] Seeing the BPM Light
http://www.ebizq.net/hot_topics/bpm/features/2705.html

[12] BPM's underpinnings, The Pi Paradigm
http://www.ebizq.net/hot_topics/bpm/features/2063.html

[13] IBM and Microsoft messing with Process Standards
http://www.ebizq.net/hot_topics/bpm/features/1472.html.

(Also see www.bpm3.com/standards for press coverage.)

[14] The Third Wave—The Humble, Yet Mighty, Business Process
http://www.ebizq.net/hot_topics/bpm/features/1763.html

[15] The Third Wave—Coordination, Coordination, Coordination
http://www.darwinmag.com/read/040103/wavecoordinate.html

Author Biographies

Howard Smith is Chief Technology Officer (Europe) of Computer Sciences Corporation (CSC) and a co-founder and current co-chair of the Business Process Management Initiative (BPMI.org). With more than 24 years in the IT industry, he is a sought-after speaker and advisor. His work in predicting and shaping technology at the intersection with business led him to take an active role in the development and application of the third wave. He is currently researching the application of business process management to corporate sustainability, innovation and growth, for which he has global research and development responsibility at CSC.

Peter Fingar is an Executive Partner with the digital strategy firm, the Greystone Group. He delivers keynotes worldwide and is author of the best-selling books, The Death of "e" and the Birth of the Real New Economy and Enterprise E-Commerce (www.mkpress.com). Over his 30-year career he has taught graduate and undergraduate computing studies and held management, technical and consulting positions with GTE Data Services, Saudi Aramco, the Technical Resource Connection division of Perot Systems and IBM Global Services, as well as serving as CIO for the University of Tampa.

Copyright © 2003, All Rights Reserved, Computer Sciences Corporation